O'REILLY®



图灵程序设计丛书

R图形化数据分析

Graphing Data with R

数据分析入门首选 // 无需编程背景

[美] John Jay Hilfiger 著

王洋洋 译


中国工信出版集团


人民邮电出版社
POSTS & TELECOM PRESS

数字版权声明

译者介绍

王洋洋

计算机硕士，狂热的数据爱好者。现为云网络安全领域大数据工程师，熟练多种编程语言、大数据技术、机器学习深度学习算法、设计模式等。业余喜欢打羽毛球、练瑜伽，古筝业余3级水平。



1. 扫描二维码观看讲解视频
2. 下载“卷积”应用并扫描封面观看AR讲解视频



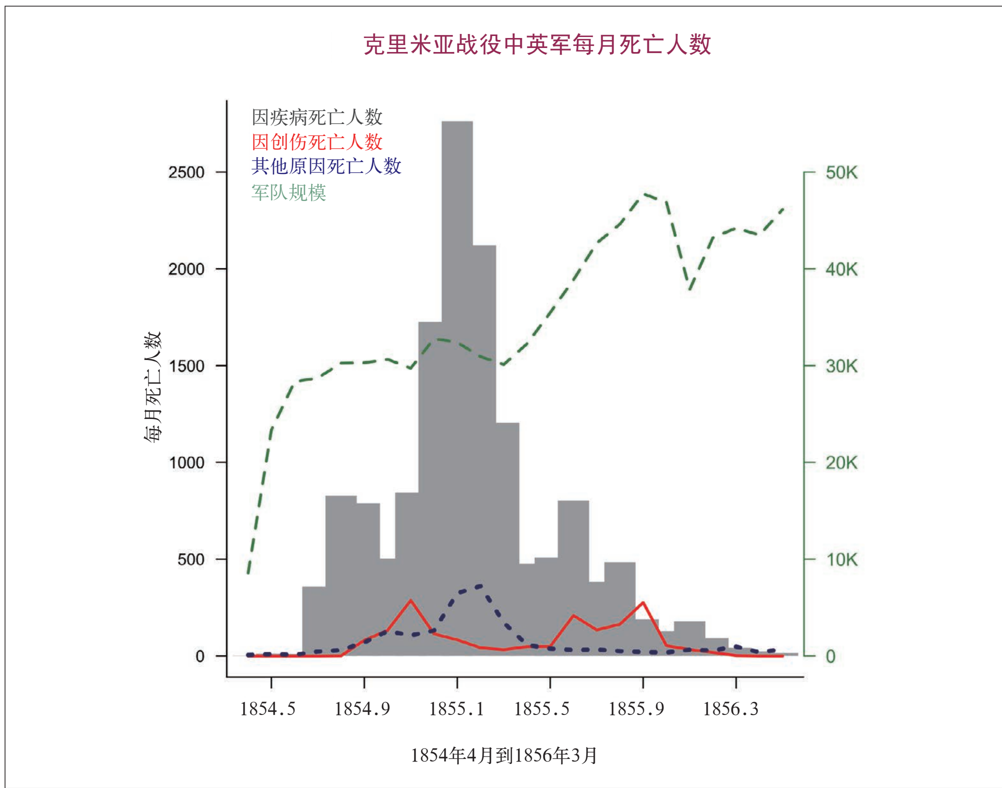


图 12-3：Nightingale 数据集的一个完整折线图

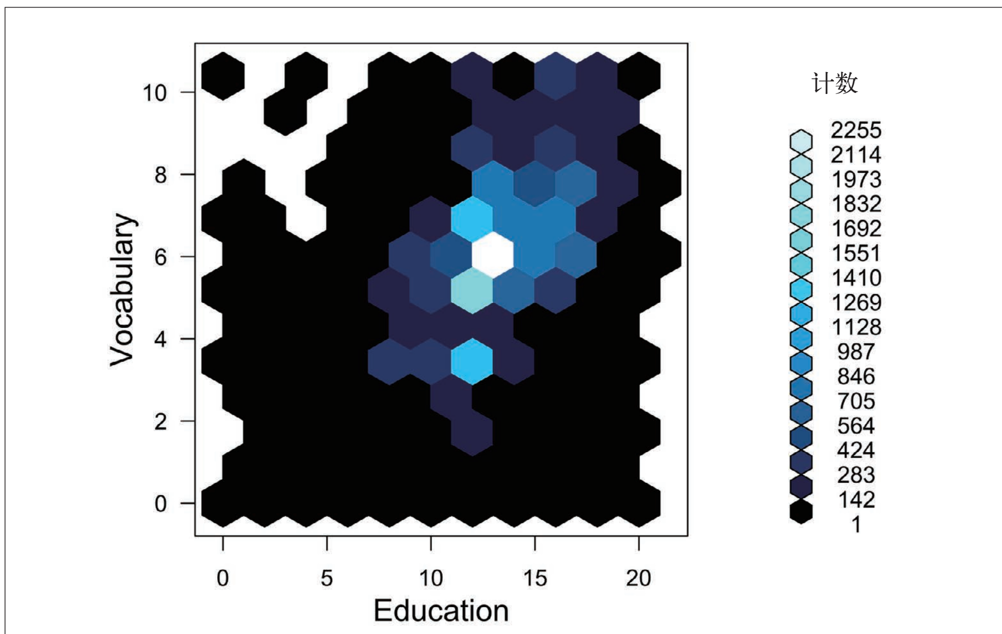


图 13-5：使用 hexbin() 函数分组的例子

	Distinct	Attend	NonChurch	StrongPct	AnnInc
Distinct	1	0.79	-0.66	0.81	-0.6
Attend	0.79	1	-0.61	0.86	-0.68
NonChurch	-0.66	-0.61	1	-0.42	0.65
StrongPct	0.81	0.86	-0.42	1	-0.61
AnnInc	-0.6	-0.68	0.65	-0.61	1

图 16-5：相关矩阵的可视化。这是一种概括或近似的散点图矩阵，由 corrplot 包中的 corrplot() 函数生成。左上图，method = "circle"；右上图，method = "color"；左下图，method= "number"；右下图，method= "ellipse"，type="lower"

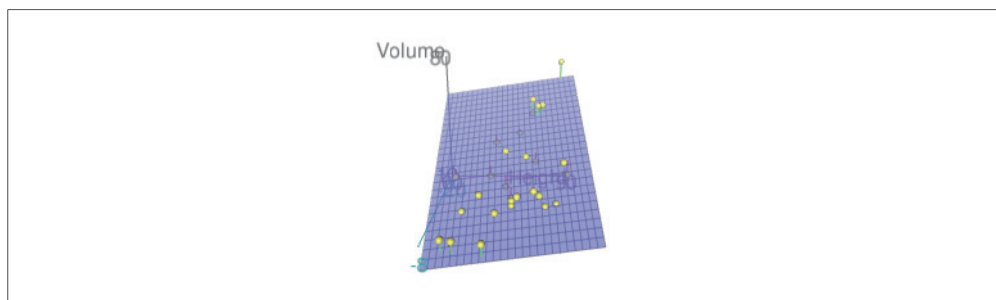


图 17-3：通过使用 car 包中的 scatter3d() 函数绘制的带有预测面板的三维散点图

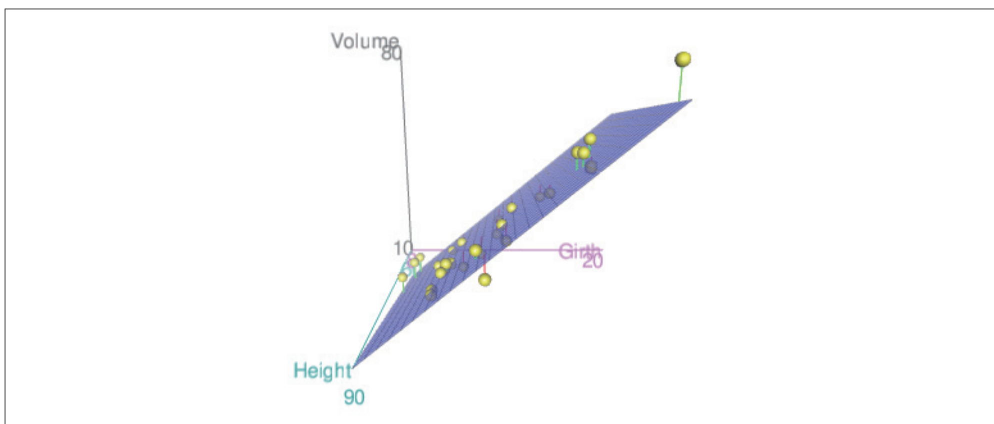


图 17-4：从另一个视角查看图 17-3。参数 `revolutions = 2` 可使其在屏幕上旋转 2 次

图 17-5：通过 `lattice` 包中的 `levelplot()` 生成的 `coalash` 数据的伪色图。任意点上 `coalash` 的总量由颜色梯度表示

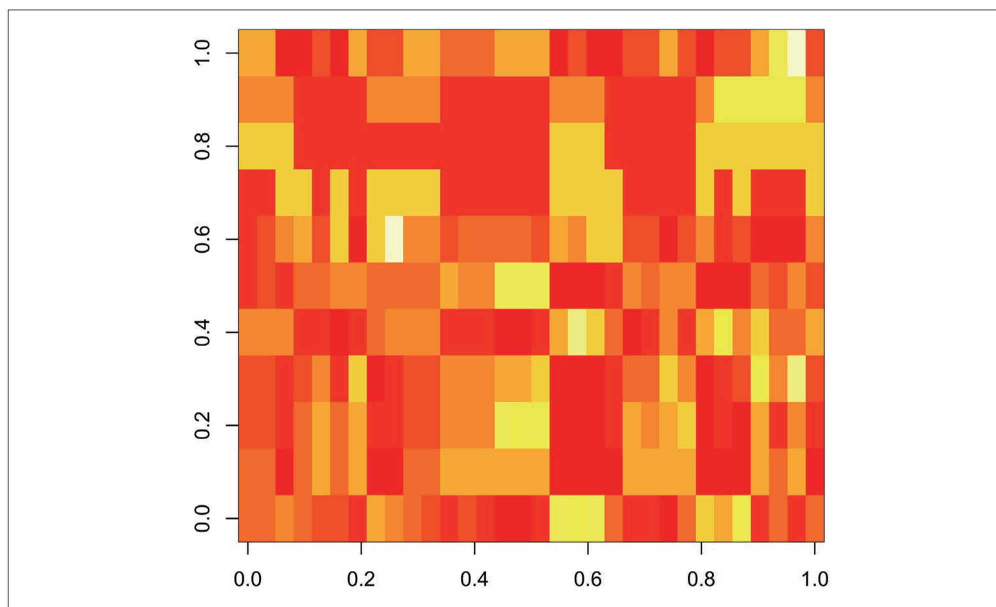


图 19-3：以默认颜色表示的 mtcars 数据集的热图

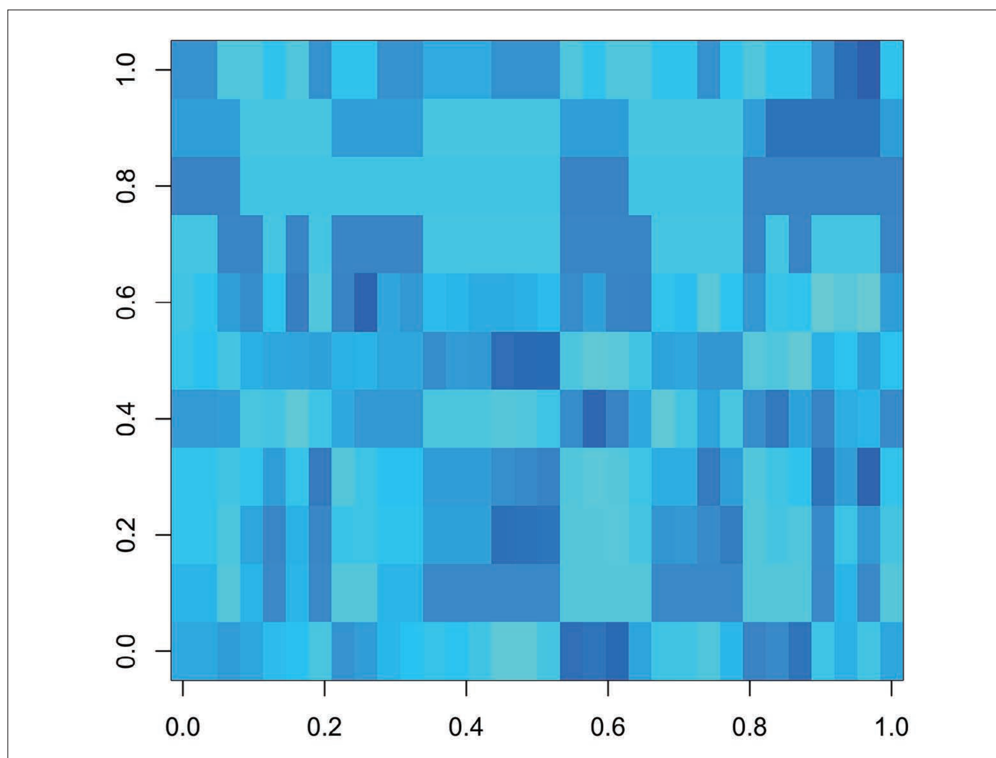


图 19-4：用一系列蓝色表示的 mtcars 数据集的热图

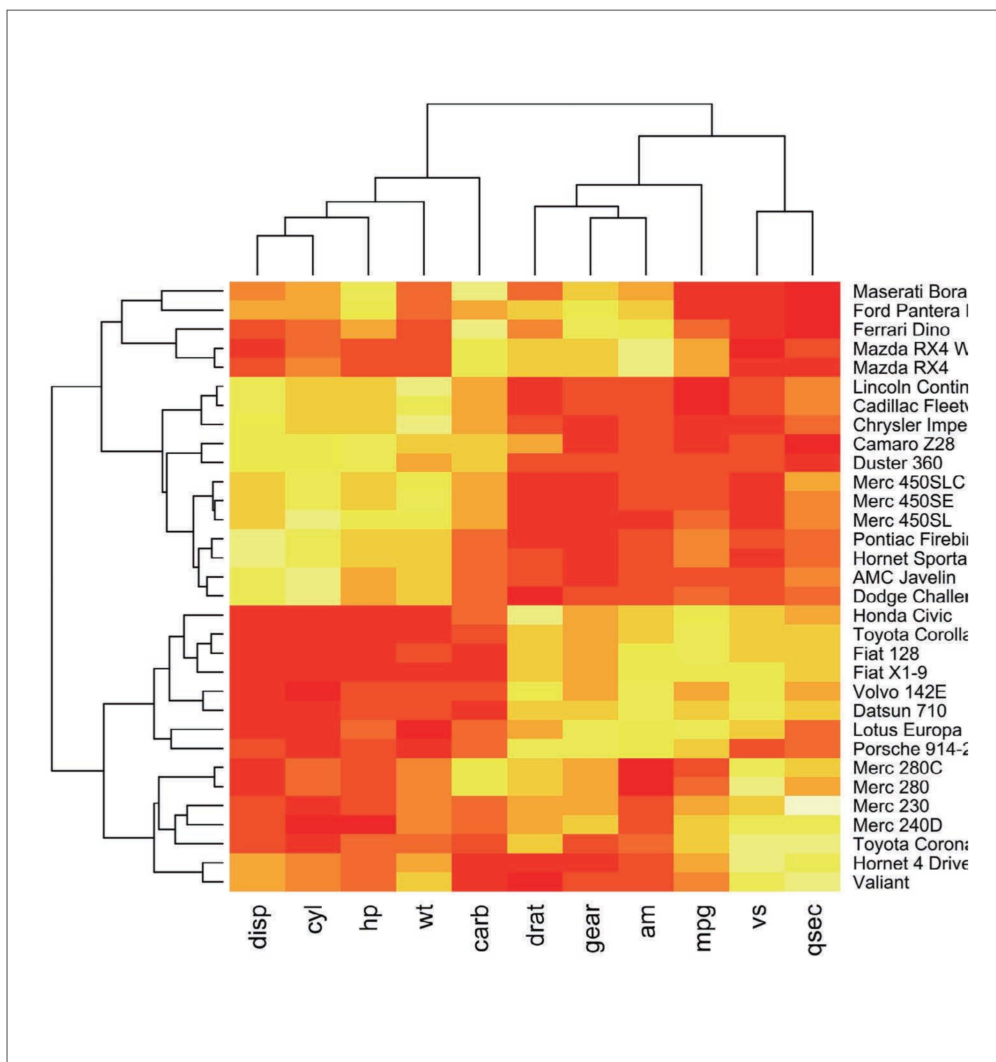


图 19-5：mtcars 中集群的热图

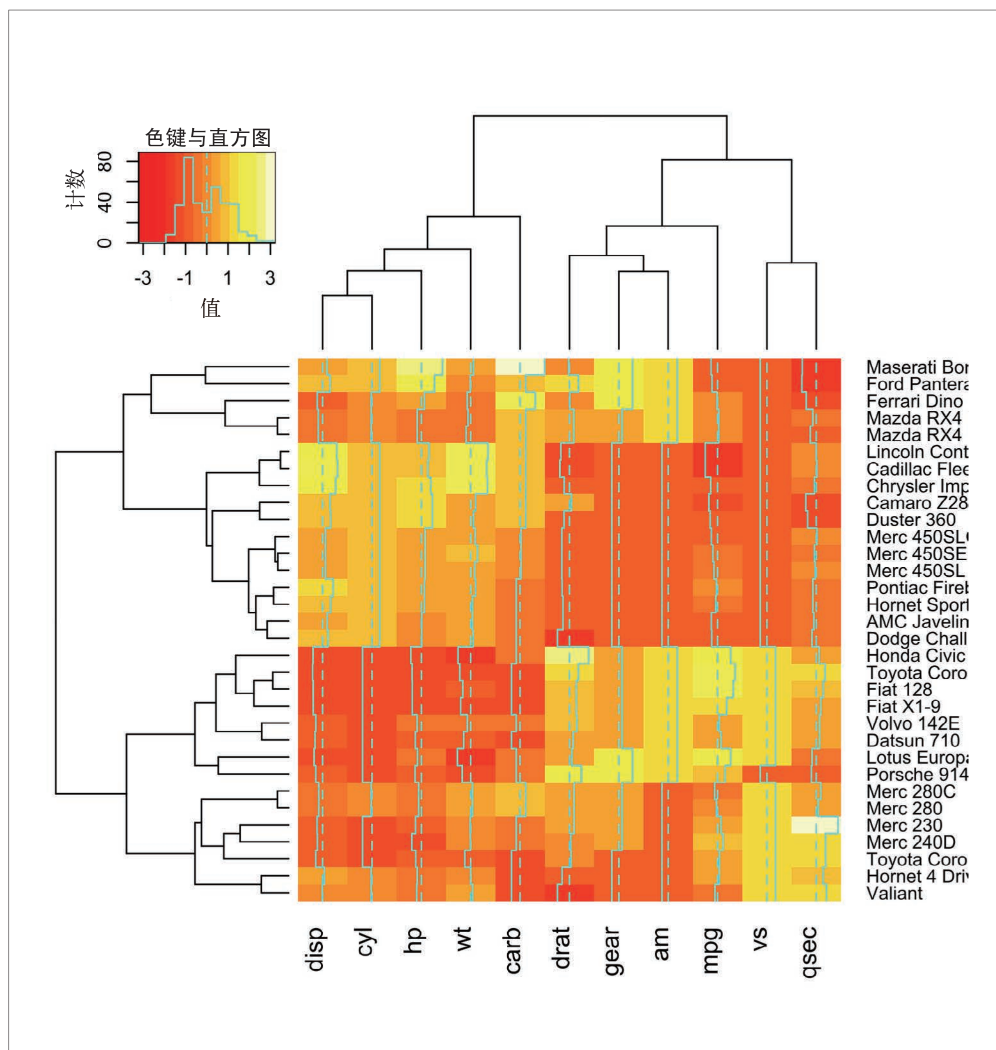


图 19-6：使用 gplots 包中的 heatmap.2() 生成的 mtcars 的热图

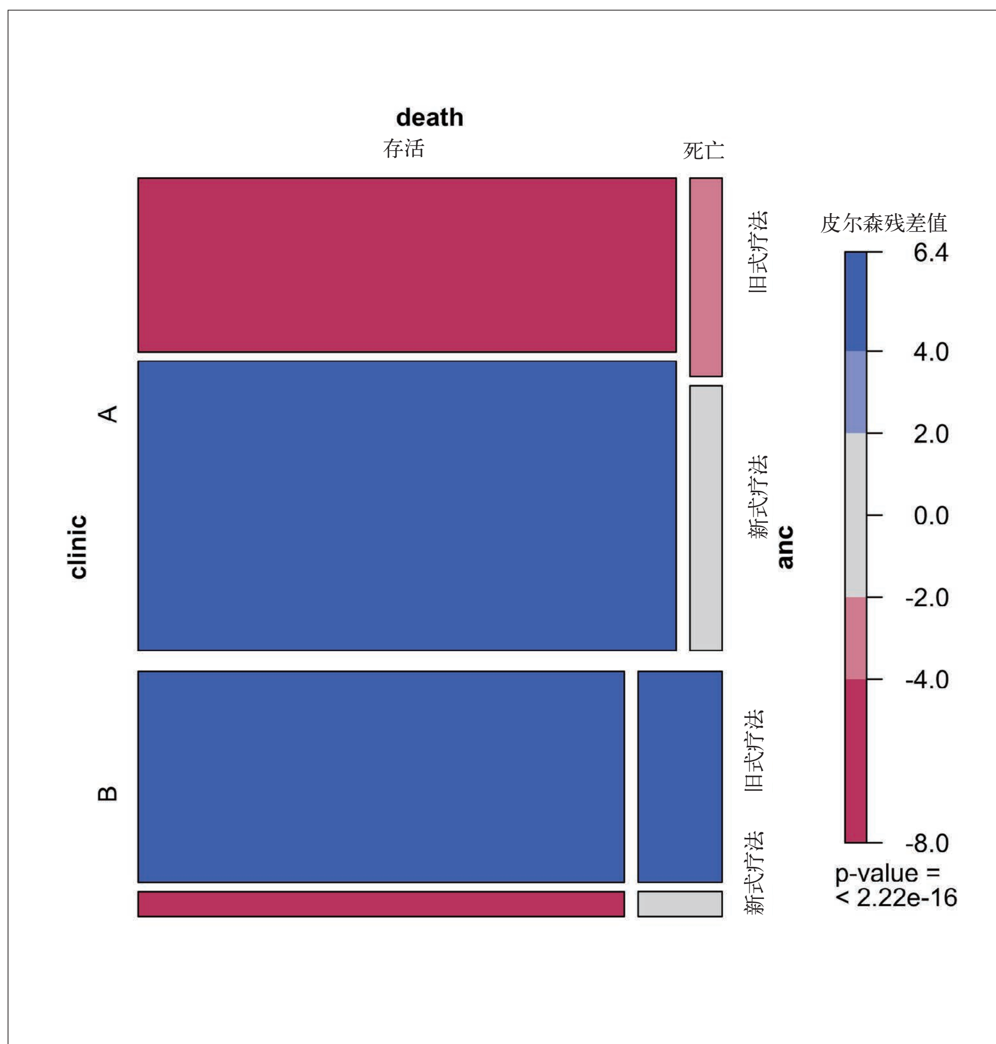


图 20-6：图 20-3 中结果的残差阴影

Named colors available in R

B-1：657 种已命名的颜色

 图灵程序设计丛书

R图形化数据分析

Graphing Data with R

[美] John Jay Hilfiger 著
王洋洋 译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo
O'Reilly Media, Inc.授权人民邮电出版社出版

人民邮电出版社
北　　京

图书在版编目（CIP）数据

R图形化数据分析 / （美）约翰•杰伊•希尔菲杰
（John Jay Hilfiger）著 ; 王洋洋译. -- 北京 : 人民
邮电出版社, 2017.8
（图灵程序设计丛书）
ISBN 978-7-115-46441-5

Ⅰ. ①R… Ⅱ. ①约… ②王… Ⅲ. ①数据处理 Ⅳ.
①TP274

中国版本图书馆CIP数据核字(2017)第180405号

内 容 提 要

本书介绍如何使用图形化的方法来分析和理解复杂的数据，该方法突出数据中重要的关联和分布趋势，并使用尽可能简单的视觉元素来呈现尽可能丰富的信息。本书重点介绍如何理解数据分析的图形元素，以及如何使用 R 生成书中涉及的各种图形。附录中列有大量参考资料，以及章节练习解答、相关 R 函数、R 包、故障排查等信息，便于读者深入学习。

本书适合任何需要数据分析和数据可视化的读者。

◆ 著　　　　[美] John Jay Hilfiger
　译　　　　王洋洋
　责任编辑　朱　巍
　执行编辑　温　雪　张　憬　回　春
　责任印制　彭志环

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号
　邮编　100164　　电子邮件　315@ptpress.com.cn
　网址　http://www.ptpress.com.cn
　北京　　　　　印刷

◆ 开本：800×1000　1/16
　印张：15.75　　　　彩插：4
　字数：372千字　　　　2017年 8 月第 1 版
　印数：1－3 000册　　　　2017年 8 月北京第 1 次印刷

著作权合同登记号　图字：01-2016-4796号

定价：69.00元

读者服务热线：(010)51095186转600　印装质量热线：(010)81055316

反盗版热线：(010)81055315

广告经营许可证：京东工商广登字 20170147 号

版权声明

O’Reilly Media, Inc.介绍

O’Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始，O’Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O’Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O’Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 *Make* 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O’Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O’Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版、在线服务或者面授课程，每一项 O’Reilly 的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O’Reilly Radar 博客有口皆碑。”

——*Wired*

“O’Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——*Business 2.0*

“O’Reilly Conference 是聚集关键思想领袖的绝对典范。”

——*CRN*

“一本 O’Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——*Irish Times*

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野，并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去，Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——*Linux Journal*

目录

第二部分 单变量图

第三部分 双变量图

第四部分 多变量图

第五部分　现在该做些什么

前言

谚语说："一图胜千言。"有时，一张图也胜过很多数据。相比口头描述细微差别或者辨别成列数字间的关系，通过观察图片或图表更容易把握数据间的复杂关系。本书主要介绍如何使用图形化方法来理解复杂的数据，该方法强调重要的关系和趋势，简化数据形式，并且使大量数据一目了然。

目标读者

任何需要分析数据和可视化数据的人，都能从本书中受益。然而，我的主要目的是使更广泛的人群理解图形数据分析，特别是那些没有太多（或任何）R 相关经验，但又需要或想要创建各种类型的图表来理解重要数据的人。这些人可能来自商业、媒体、平面艺术、社会科学或者健康科学领域，真的需要分析数据，但可能并没有高等数学和计算机编程的背景。虽然本书专为自学设计，但也可作为初中级统计课程或研究的补充材料。

本书使用的工具是 R。这不是一本关于 R 的内容全面的教材。许多计算机课程和图书都试图告诉你借助一种语言或工具可能做的每一件事。对于曾经想按此方式学习的大多数人来说，这种方式令人感到十分烦恼和无聊。本书将把重点放在理解数据分析的图形元素和如何使用 R 生成本书讨论的各种图形，也将展示如何使用 R 的一些内置资源来获得帮助，很多其他内容则留给你继续探究。你应该有台可用的计算机，用它可轻松完成一些工作，如发送电子邮件、浏览互联网，或者使用文字处理软件、电子表格等应用程序。熟悉基本的统计知识有利于理解本书的一些主题，但对于大多数主题，这并不是必需的。

为什么选择R

小数据量的图表可以手工制作，但是利用计算机技术会更高效、准确地分析数据，生成有吸引力的图形。对于大批量数据来说，手工处理实际上是不可能的。而运用计算机软件，即使是针对非常大的数据量，也可以生成复杂的图形。

实际上，开源软件已经实现了该技术，只要拥有一台计算机。“开源”指的是所有人均可获取项目的源代码，可检查、使用、自由修改或增加源代码。

开源软件产品可提供免费下载给任何有需要的人。或许你会怀疑免费的东西质量不高，但我向你保证，一些自由软件遵循了最高的专业标准。

本书选用的 R 语言是一种编程语言，是一个统计、数学和绘图程序集合，已经被世界各地数百万人使用，包括科学、商业和媒体等领域的许多专业人士。在网站、主要报纸和其他出版物上，你可能见过由 R 制作的图形。你也将能够制作出这种专业的数据图表，因为 R 可运行在 Windows、Mac 或 Linux 操作系统上，而现在的 PC 和笔记本无非就这几类系统！

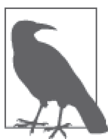
如何使用本书

要想从本书获益，你需要动手制作大量图表。为此，阅读本书时，你最好坐在计算机前操作书中给出的所有命令。而且为帮助你提升水平，许多章节除示例以外还提供了练习，比如优化示例代码或将不同的数据集制成另外一张图。最好先做完这些练习再进入下一主题。

排版约定

本书使用了下列排版约定。

- 楷体
  表示新术语。
- 等宽字体（`constant width`）
  表示程序片段，以及正文中出现的变量、函数名、数据库、数据类型、环境变量、语句和关键字等。
- 加粗等宽字体（**`constant width bold`**）
  表示应该由用户输入的命令或其他文本。
- 斜体等宽字体（*`constant width italic`*）
  表示应该由用户输入的值或根据上下文确定的值替换的文本。



该图标表示一般注释。

代码示例的使用

本书会帮你完成工作。一般来说，如果本书提供了示例代码，你可以把它用在你的程序或文档中。除非你使用了很大一部分代码，否则无需联系我们获得许可。比如，用本书的几个代码片段写一个程序就无需获得许可，而销售或分发 O'Reilly 图书的示例光盘则需要获得许可；引用本书中的示例代码回答问题无需获得许可，而将书中大量的代码放到你的产品文档中则需要获得许可。

我们很希望但并不强制要求你在引用本书内容时加上引用说明。引用说明一般包括书名、作者、出版社和 ISBN。比如："*Graphing Data with R* by John Jay Hilfiger (O'Reilly). Copyright 2016 John Jay Hilfiger, 978-1-491-92261-3."

如果你觉得自己对示例代码的用法超出了上述许可的范围，欢迎你通过 permissions@oreilly.com 与我们联系。

Safari® Books Online



Safari Books Online（http://www.safaribooksonline.com）是应运而生的数字图书馆。它同时以图书和视频的形式出版世界顶级技术和商务作家的专业作品。技术专家、软件开发人员、Web 设计师、商务人士和创意专家等，在开展调研、解决问题、学习和认证培训时，都将 Safari Books Online 视作获取资料的首选渠道。

对于组织团体、政府机构和个人，Safari Books Online 提供各种产品组合和灵活的定价策略。用户可通过一个功能完备的数据库检索系统访问 O'Reilly Media、Prentice Hall Professional、Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett、Course Technology 以及其他几十家出版社的上千种图书、培训视频和正式出版之前的书稿。要了解 Safari Books Online 的更多信息，我们网上见。

联系我们

请把对本书的评价和问题发给出版社。

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

　　北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室（100035）
　　奥莱利技术咨询（北京）有限公司

你还可以发送电子邮件到 bookquestions@oreilly.com。

勘误、示例和其他信息可到 http://www.oreilly.com/catalog/0636920038382.do 上获取。

欲了解本社图书、课程、会议和新闻等更多信息，请访问我们的网站 http://www.oreilly.com。

我们在 Facebook 的地址如下：http://facebook.com/oreilly。

请关注我们的 Twitter 动态：http://twitter.com/oreillymedia。

我们的 YouTube 视频地址如下：http://www.youtube.com/oreillymedia。

致谢

在很多人的帮助下，我完成了这本书。首先，妻子 Karen 在我整个写作过程中给予的耐心、理解和鼓励，对我完成本书至关重要。我们的儿子 Eric 和女儿 Kristen 读了第 1 章后，给出了相当直接的评价，使我感到羞愧但很有帮助。担纲技术审校的 Dr. Peter Bajorski、Sarah Boslaugh 和 Philipp K. Janert 的见解、纠正和建议是很宝贵的。本书编辑 Shannon Cutt 非常积极能干，不仅在写作上提供帮助，而且在准备手稿的所有技术和操作细节上提供帮助。我不知道竟有这么多工作需要做！最后，O’Reilly 团队做了所有你看得到和看不到的事情，这一切对于生产高质量的图书至关重要，他们是如此令人尊敬。感谢所有人。

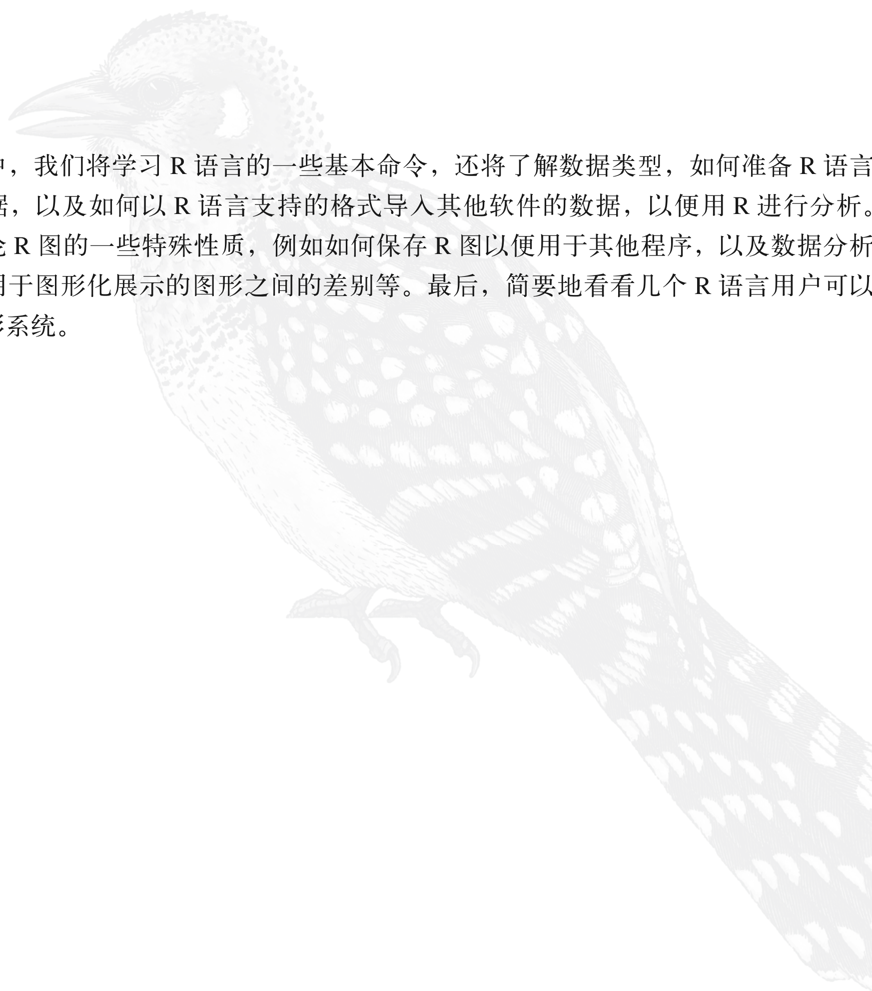
电子书

扫描如下二维码，即可购买本书电子版。



第一部分

开始使用R



在本部分中，我们将学习 R 语言的一些基本命令，还将了解数据类型，如何准备 R 语言使用的数据，以及如何以 R 语言支持的格式导入其他软件的数据，以便用 R 进行分析。之后将讨论 R 图的一些特殊性质，例如如何保存 R 图以便用于其他程序，以及数据分析的图形和用于图形化展示的图形之间的差别等。最后，简要地看看几个 R 语言用户可以使用的图形系统。

第1章 R基础

1.1 下载软件

首先需要下载免费的R软件，并安装在你的计算机上。启动计算机，打开Web浏览器，通过网址http://www.r-project.org访问统计计算的R项目（R Project for Statistical Computing）。点击download R，然后选一个距离你所在地理位置较近的镜像站点。（R软件存储在世界各地的许多计算机上，而不是一台计算机上。因为它们都包含相同的文件，看起来一样，所以被称为“镜像”网站。你可以选择其中任何一台计算机。）点击网站地址，将打开一个页面，可以根据你的操作系统选择R的版本。如果你的计算机可以运行最新版本的R——3.0或更高版本——那再好不过了。然而，如果你的计算机用了几年了，不能运行最新的版本，那就选计算机可以运行的最新版本。也许会和本书中的例子有一些小的差异，但应该可运行大多数例子。

按照说明，在短时间内就可安装好R。这是基础R（base R），但R有数千个（这不是夸张）插件“包”。完成R的基础安装后，你可以免费下载插件来扩展R的功能。根据你的需要，也可能不需要添加任何插件，但是你可能会很惊奇地发现一些你想象不到但必须拥有的功能。

1.2 尝试一些简单的任务

若你使用的是Windows或OS X系统，单击桌面上的“R”图标即可启动R。若使用的是Linux或OS X系统，在终端窗口输入命令R可打开控制台（console）。在控制台窗口输入命令，可看到许多命令的结果，尽管在大多数情况下，创建图形的命令将打开一个新窗口

来展示结果图。R 可接受命令时，会显示一个大于符号（>）作为提示符。R 最简单的应用是计算器。在提示符后，输入一个你想要获得答案的数学表达式：

```
>12/4
[1] 3
>
```

此处，我们请求“12 除以 4”。R 返回“3”，然后显示另一个提示符“>”，表明可以输入下一个命令。返回值前的 [1] 是一个*索引*（index），在本例中，它只是表明返回值从*向量*（vector）中的第一个数开始。本例中只有一个值，但有时会有多个值，所以了解数据集合从哪里开始会有帮助。如果你不理解索引，现在也不用担心；看到更多的例子后，你将更清楚。除法符号（/）叫作*操作符*（operator）。表 1-1 给出了标准算术运算符的符号。

表1-1：R算术运算符

运算符	运算	举例
+	加法	3 + 4 = 7 或 3+4（即无空格）
-	减法	5 - 2 = 3
*	乘法	100*2.5=250
/	除法	20/5= 4
^ 或 **	幂	3^2 = 9 或 3**2 = 9
%%	取余	5%%2 = 1（5/2=2 余 1）
%/%	除法，向下取整	5%/% = 2（5/2=2.5，向下取整等于 2）

你可以像在普通算术中那样使用括号，以表明操作的执行顺序：

```
> (4/2)+1
[1] 3
> 4/(2+1)
[1] 1.333333
```

尝试另一个例子：

```
> sqrt(57)
[1] 7.549834
```

这次通过*函数*（function）来完成运算，本例用的是 sqrt() 函数。表 1-2 列出了一些常用的算术函数。

表1-2：常用的R数学函数

函数	运算	函数	运算	函数	运算
cos()	余弦	exp()	指数函数	max()	最大值
sin()	正弦	sum()	求和	var()	方差
tan()	正切	mean()	均值	sd()	标准差
sqrt()	平方根	median()	中间值		
log()	对数	min()	最小值		

调用函数时可带参数（argument）。参数是一种修饰符，与函数一起使用时可以用更多特殊的方式请求R。因此，可能会请求计算特定数字的和，而不是单单请求sum函数；或者，你可以使用参数来指定线的颜色或宽度，而不是简单地在图上画一条线。单个或多个参数，必须在函数名后用括号括起来。当使用函数或任何R命令时，如果需要帮助，可使用如下方式寻求帮助：

```
> help(sum)
```

R 将打开一个新窗口，显示要查找的指定函数及其参数的信息。如下命令是一个快捷方式，可得到完全相同的响应：

```
> ?sum
```

请注意，R 是区分大小写的，所以“help”和“Help”是不同的！然而，空格无关紧要，因此上面的命令也可写为：

```
> ? sum
```

有时，函数中只有一个参数，比如 sqrt() 的示例。其他情况下，一个函数作用于一组数字，称为向量，如下所示：

```
> sum(3,2,1,4)
[1] 10
```

本例使用 sum() 函数计算 3、2、1、4 的和。但不可能总是像这个例子一样，把所有的值写入函数表达式。因此，通常需要先创建向量，如下所示：

```
> x1 <- c(1,2,3,4)
```

输入这个命令后，什么也没发生！你确实看到什么也没有发生。一旦出现由“<”和“-”两个符号组成的特殊操作符，操作符右边的值就会赋给左边的变量。（R 的较新版本允许使用“=”号来完成赋值。第 1 章以后，我们也将使用这种较简单的形式。）在这种情况下，创建一个新的向量，用户称其为 x1。R 是一种面向对象语言（object-oriented language），向量 x1 是工作区中的一个对象（object）。

什么是“对象”

将对象想象为一个盒子，其中装满了彼此关联的项。这些项可以是简单的数字、名称、统计分析的结果、这些项或其他项的组合。对象有助于以结构化的方式组织事物，将彼此相关的东西封装在同一个盒子里，无关的东西封装在其他盒子里；对象可以告诉 R 在其中有哪些项，以便 R 合理地操作特定对象中的项。向量是一类包含相同类型数据（都是数字或者都是字母）的对象。对象可以包含其他对象。毕竟，你可以把一个盒子放入一个更大的盒子里。因此，可以把一个或多个向量放入一个**数据框**（data frame），数据框是另一种类型的对象。输入命令 **ls()**，可以查看当前工作区有哪些对象。

创建一个新的向量，需要在向量数据的括号前输入字母“c”。创建后输入以下命令看看会发生什么：

```
> x1
```

已经用 x1 为名保存了一组数字 1、2、3、4。输入向量的名字就可打印出 x1 的值。你可以随时请求 R 对该向量执行各种操作。例如，输入如下命令：

```
> mean(x1)
```

将返回（return）向量 x1 中数字的均值，打印到屏幕上。利用表 1 - 2 中的其他运算符可以了解 R 可以做的其他事情。

创建另一个对象，这次创建仅有一个数字的对象：

```
> pt <- 3.14
```

任何时候想获得当前工作区中所有对象的列表，可使用如下命令：

```
> ls()
```

并且，在新的计算中可以使用任何一个或所有对象：

```
> newvar <- pi*x1
```

这样就又定义了一个名为 newvar 的对象。

1.3 用户界面

目前为止看到的示例都是命令行指令（command-line instruction），也就是说直接输入命令告诉 R 做什么。这不是与 R 交互的唯一方法，R 的基础安装也具备图形用户界面（graphical user interface，GUI）功能。GUI 指的是点击式的图形界面，你很可能已在其他应用程序中见过。但问题是，安装的每类操作系统——Windows、OS X 和 Linux——的 GUI 功能是存在差别的。相比其他系统，OS X 的 GUI 更友好一点，也许你很快就能知道，也更喜欢以这种方式发出很多命令。无论使用哪款操作系统，控制台窗口的顶部均有一个菜单。在输入重要的数据之前，可以简单试一下，看看有哪些点击式操作可用。

由于三种操作系统（Windows、OS X 和 Linux）的 R 命令行界面是相同的，因而本书使用命令行界面（command line interface）。这样只需一种解释，你就可以轻松地从一台计算机转移到另一台。列举代码（code），即一系列命令行，比试图解释每一个选择按钮和鼠标点击要容易得多。此外，使用命令行学习 R，有助于你更好地理解软件的逻辑。最后，命令语言比界面点击更精确，并能给予用户更强的操控力。

1.4 安装包：GUI界面

无论你使用的是什么操作系统，都可以下载一个免费的“前端”（frontend）程序，它将提供一个 GUI。有几个可用的“前端”程序。在对 R 有了更多的了解，并欣赏了其强大的用途之后，你可能就会尝试一个 GUI 接口了。例如，早些时候我提到有大量包可用，你可以将其添加到 R，其中一个设计不错的 GUI 名为 R Commander。在联网的情况下，用下面的命令：

```
> install.packages("Rcmdr", dependencies=TRUE)
```

R 会下载这个包及 R Commander 正常运行所依赖的其他包。包将永久保存在你的计算机上，所以不需要再重新安装。每次要使用 R Commander，都需要按如下方式加载（load）包：

```
> library(Rcmdr)
```

每个人的喜好不同。有些人认为命令语言很好；另一些人则不喜欢 R 的命令行界面，觉得正是 R Commander 使 R 成为他们最喜欢的计算机工具。使用 R Commander 能生成本书中的许多图，但不能生成所有的图。如果你想尝试 R Commander，可在附录 C 中找到更多信息。

检索所有可用的包，可使用如下命令：

```
> available.packages()
```

在 CRAN 任务视图中，你可以学到更多关于这些包的主题，地址为：http://cran.r-project.org/web/views/。

访问 http://cran.r-project.org/web/packages/available_packages_by_name.html，可以看到所有按包名排列的包的列表。

获得刚刚下载的包的帮助，可输入以下命令：

```
> library(help=Rcmdr)
```

错误信息

当输入一个错误命令时，将不会看到预期的结果，而是看到一条错误信息，它不一定能提供帮助！附录 G 就如何处理最可能出现的错误类型提供了指导。

1.5 数据结构

你可以把数据写入各种结构的对象。我们已经使用过向量这一类型的结构。可以把向量看作一维的一行元素或一列元素。向量可以包含任意多的元素，计算机的内存可以容纳

多少，向量就可容纳多少。向量中的元素类型可以是*数值型*（numeric），或者包含字母、数字和特殊字符的*字符型*（character），或者包含 TRUE（真）或 FALSE（假）值的*逻辑型*（logical）。向量的所有元素必须是相同的类型。下面是一些创建向量的例子：

```
> x <- c(14,6,7,5.1,-8)  # 数值型
> name <- c("Lou", "Mary", "Rhoda", "Ted")   # 字符型/需要引号
> test <- c(TRUE, TRUE, TRUE, FALSE, TRUE)   # 逻辑型/需要大写
```



符号 # 后面是注释，供我们阅读的信息或注意事项，但 R 会忽视这些注释（作为一名音乐家，我更喜欢称这个符号为*升半音号*）。给代码写注释是一个好习惯，以后回头再看代码时，这有助于你想起为什么做某一件事，并帮你解决问题或根据一个好主意来扩展代码。阅读本书中 R 示例代码的注释，也是个不错的主意。

数据框（data frame）是我们将使用的主要结构。它是一个二维的对象，有行和列。你可以把它视为其中有列向量的盒子，或一个有行和列的*矩形数据集*（rectangular dataset）。为了更好地理解，参见下一节的样本数据集及把二氧化碳排放数据读取到 R 的练习。数据框可以包含所有相同类型的列向量或任何类型的组合。

R 还有其他数据结构，比如矩阵、数组、列表，这里不做讨论。

可以使用 str() 方法查看给定对象的*结构*（structure）：

```
> str(x)
 num [1:4] 14  6.7  5.1  -8

> str(name)
 chr [1:4] "Lou" "Mary" "Rhoda" "Ted"

> str(test)
 logi [1:5] TRUE TRUE TRUE FALSE TRUE
```

1.6 样本数据集

基础 R 包含一些样本数据集，这将有助于我们通过实例去了解图形工具。要想知道你的计算机上有哪些可用数据集，可输入如下命令查看：

```
> data()
```

确保空括号放在命令之后，否则你将得不到预期的结果。还有很多可用的数据集。几乎所有附加包均有样本数据集。只需使用帮助命令，就可以查看基础 R 包或者已经下载的包中特定数据集的描述。例如，要获得 airquality 数据集的信息——简介、来源、参考文献等，可输入以下命令：

```
> ?airquality
```

通过使用如下命令，可查看数据集中前 6 组观测值：

```
> head(airquality)

  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

该数据集是个数据框，有 153 行数据，每行代表了一天中空气质量的测量值（臭氧、R 太阳能和风力等）。head() 命令默认打印出变量名及前 6 行数据，以便我们可以查看数据情况。若要查看不同行数的数据，比如 25 行，可使用如下方式：

```
> head(airquality,25)
```

若要查看数据集的最后 4 行数据，可输入如下命令：

```
> tail(airquality,4)
```

每行数据有行号和 6 个变量的值，这些值是一天的测量值。第一行或者说第一天的值，有 1、41、190、7.4、67、5、1。第一个变量是臭氧，前 6 天的值为 41、36、12、18、NA、28。这是一个关于*矩形数据集*或者*平面文件*（flat file）的例子。大多数统计分析程序需要这种格式的数据。

注意，在数据集的数据中，你可能会看到 NA。NA 是标准的 R 符号，表示“不可用”或“丢失”。可以用多种方式处理这些值，一种方式是删除存在一个或多个缺失值的行，并和其他所有行做计算；另一种方式是不做计算，并返回一条错误信息。一些程序可以让用户指定使用哪种方式。也可以为缺失值*插补*（impute），即估值，并在计算中使用估值。处理缺失值是一个复杂且有争议的话题，不能掉以轻心。关于如何处理 R 的缺失值，Kabacoff（2011）中有一章做了很好的介绍。

有两种访问数据的方式。第一种方法是使用 attach() 命令，输入一些带变量名的命令，然后输入 detach() 命令，如下例所示：

```
> attach(airquality)
> table (Temp)              # 获得Temp值的数量
> mean (Temp)               # 寻找Temp的均值
> plot(Wind,Temp)           # 绘制Wind和Temp的散点图
> detach(airquality)
```

这种方法的优点是，如果要执行多个步骤，没有必要一遍又一遍地输入数据集名。第二种方法是用数据集名和变量名的组合，并用美元符号（$）分隔，来指定任何你想做的分析。例如，如果想执行如下命令：

```
> attach(airquality)
> plot(Wind,Temp)
> detach(airquality)
```

就可以执行如下的等效代码：

```
> plot(airquality$Wind,airquality$Temp)
```

这种方法的优点是，如果连续请求多个数据集，就不用使用多个 `attach` 和 `detach` 语句。

1.7 工作目录

使用 R 时，经常需要把数据从文件读到 R，或者将 R 数据写入文件。例如，你可能有些数据是由电子表格、SAS 和 SPSS 等统计软件包或文本编辑器创建的，想要用 R 分析。或者你常会创建一个 R 数据集，保存并再次使用它。这些文件必须存储在计算机的文件结构中。对于每个读写操作，可以指定一个明确（通常很长）的路径，指向包含你想要读取的数据的文件或你想要写入数据的地方。这样可能比较麻烦，因此 R 有*工作目录*（working directory），即文件的默认位置。换句话说，如果不告诉 R 在哪里找某个特定的文件，它会默认在工作目录中查找。同样，如果不指定保存数据的位置，R 将自动写到工作目录。可以用如下命令查看当前工作目录：

```
> getwd()
```

假设获得的返回结果如下（当然实际结果是完全不同的）：

```
[1] "/Users/yourname/Desktop/"
```

若不指定其他位置，R 则会在一长串名称的最后一个文件夹（即右边最后一个名字）查找文件和写入文件。使用 `setwd()` 命令，可以改变工作目录。为使用 R，或者专门为了本书的练习，你可能想创建一个新文件夹，其名字要清楚表明目的，如 R folder 或 R graphical data。假设你已经在 Desktop 目录内创建了一个名为 R things 的文件夹，可以输入以下命令：

```
> setwd("/Users/yourname/Desktop/R things")
```

这样，R 将把 R things 目录作为你的工作目录，直到下次使用 `setwd()` 命令，或输入 **`q()`**（quit）关闭 R。如果不想每次启动 R 时都设置工作目录，在 1.9 节中学习如何实现这一点。

1.8 将数据导入R

现在，你已经知道怎么使用各种 R 包的样例数据集了。这是学习使用 R 的丰富资源，但是，你学习 R 是因为想对自己的数据做图形分析。选择哪种方法把数据导入 R，取决于以

下几个因素：

- 数据集有多大
- 数据是否已经以某种数据文件的形式存在
- 使用 R 之外的其他工具的舒适度
- 你得在数据输入上投入多少时间
- 自身的痛苦阈值

> **初学者注意**
>
> 接下来三节展示了多种输入数据的方法。如果你是一个初学者，觉得这几节内容太难，可以先读 1.8.1 节，然后尝试解决一个简单的输入数据的问题，如本章最后的练习 1-4；稍后再返回到“使用数据编辑器”（1.8.2 节）和“从外部文件读取”（1.8.3 节）。事实上，做完练习 1-4 之后，可以直接进入第 3 章，然后当需要相关信息时，再开始阅读 1.8.2 节，直到读完第 2 章。

如果你对数据输入不是特别感兴趣，而是希望使用已经创建好的数据集，比如电子表格、统计软件包数据集、ASCII 文件或其他类型的数据文件，可以略过本节余下部分，直接参考附录 E 来查阅你感兴趣的数据文件类型。

1.8.1 命令行输入

将数据输入 R 的最直接的方法，是从命令行输入表达式，如前文操作过的——创建向量。如果你需要的是分析一个或几个特别短的向量，这种方法最简单。

练习1-1

Backblaze 是一家数据备份公司，运行着大约 25 000 个磁盘驱动器并报告硬盘存活率（单位为 %）。下列数据显示了驱动器的年度存活率（数据来源于 https://www.backblaze.com/blog/how-long-do-disk-drives-last/）：

```
year  rate
1   94    #（比如,一年之后,94%的驱动仍在工作）
2   92
3   90
4   80
```

使用如下命令创建两个向量：

```
> year <- c(1,2,3,4)
> rate <- c(94,92,90,80)
```

确保以正确的顺序输入数字。例如，如果 `1` 是 `year` 向量的第一个值，那么 `94` 必须是 `rate` 向量的第一个值，以此类推。可使用下面这个命令分析这两个向量之间的关系：

```
> plot(year,rate)
```

大多数图形命令会打开一个新窗口。如果你开启多个应用程序，可能会丢失当前的命令并被迫去寻找它。

上面代码片段中的 `plot` 语句调用 `plot()` 函数，令其对 `year` 和 `rate` 两个参数进行分析。刚刚绘制的是一个简单的图表，但用 R 也可制作非常复杂精美的图表。图 1-1 中的右图展示了一些自定义基础图的方法。在本书中我们将介绍许多这样的选项。输入 **`?plot`** 命令，查看可用的选项列表。

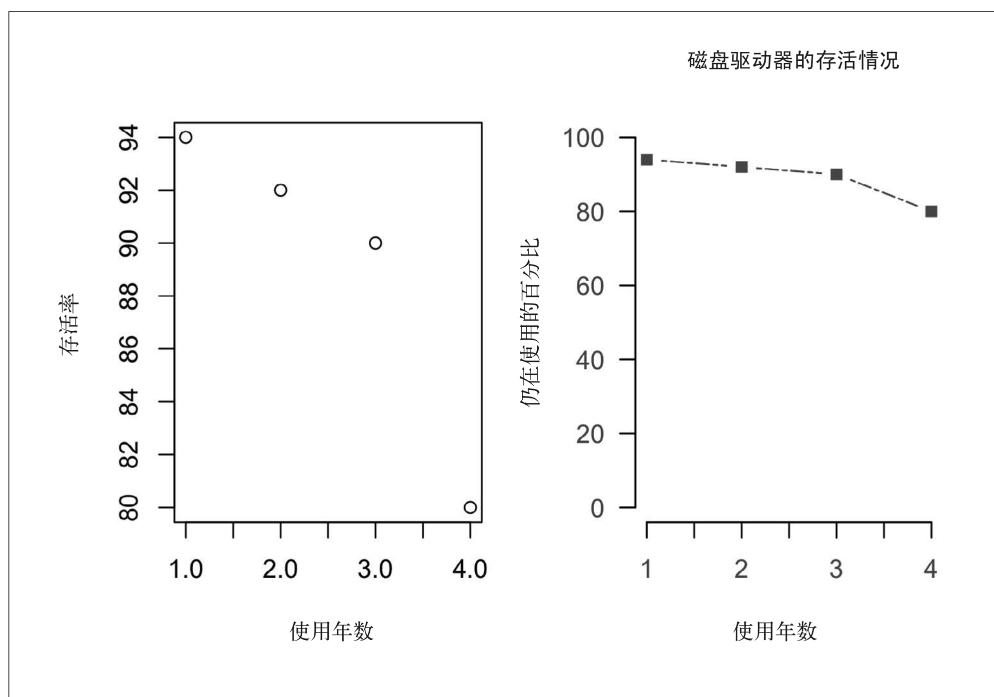


图 1-1：左图表示磁盘驱动器存活率与使用年数，是由简单的命令 `plot(year,rate)` 生成的。右图是自定义的，需要很多的选择。你发现了多少差异？

可以把 `year` 和 `rate` 两个向量合并到一个新的数据框 `mydata`，如下所示：

```
> mydata <- data.frame(year, rate)
```

1.8.2 使用数据编辑器

如果数据只是稍微复杂或者数据量稍大，可以使用 R 控制台中简单的数据编辑器。即使不使用这种方式录入数据，了解一下编辑器也不错，因为有朝一日你可能需要解决 R 工作区中的临时问题数据点。我认为，对大多数人来说，努力尝试使用数据编辑器输入数据是不

必要的。阅读本节了解一些术语，并看一下如何保存文件即可。你可能更愿意使用最喜欢的电子表格来录入数据，但是如果没有电子表格，可能就需要使用编辑器。阅读 1.8.3 节来了解如何从电子表格读取数据到 R。

练习1-2

表 1-3 中给出的数据（来自美国能源信息管理局）是近 8 年间全球二氧化碳的排放量。你将使用 R 内置的数据编辑器将它输入 R。先来看看这个数据集的内容。

表1-3：按世界地区划分，从能源使用角度的二氧化碳人均排放量（以人均二氧化碳吨数计）

年份	北美	中南美	欧洲	欧亚大陆	中东	非洲	亚洲/大洋洲
2004	16.2	2.4	7.9	8.5	7.1	1.1	2.7
2005	16.2	2.5	7.9	8.5	7.6	1.2	2.9
2006	15.9	2.5	7.9	8.7	7.7	1.1	3.1
2007	15.9	2.6	7.8	8.6	7.6	1.1	3.2
2008	15.4	2.6	7.7	8.9	7.9	1.2	3.3
2009	14.2	2.6	7.1	8	8.3	1.1	3.5
2010	14.5	2.7	7.2	8.4	8.4	1.1	3.6

（来源：http://1.usa.gov/1R6sj99）

表 1-3 中第一行是*标题*（header）信息，命名记录的每个变量。每行包含一年中收集到的所有信息。每一行被称为一个*统计单元*（statistical unit）。社会学家通常把一行称作一个*实例*（case），而自然科学家的一行很多时候指的是*观测*（observation）。计算机专业人士通常把一行称为一条*记录*（record）。每一列为一个*变量*，在计算机科学中为一个*字段*（field）。排放数据集有 7 行（观测值），共 8 个变量：年份及所研究的 7 个地区各自的排放总量。

编辑器看起来像电子表格，并拥有一些良好的电子表格的特性，但不如 Excel 或 Numbers 使用方便，而且一不小心很容易丢失变更信息。一开始，需要选择对象名称，用该名称定义一个新的数据框。实现这一点有几种方法。我觉得最安全的方法是为每个变量命名、确定其类型并指定行数。代码如下：

```
> emissions <- data.frame(Year=numeric(7),N_Amer = numeric(7),
CS_Amer=numeric(7), Europe=numeric(7),Eurasia=numeric(7),
Mid_East=numeric(7),Africa=numeric(7), Asia_Oceania=numeric(7))
```

以上代码创建了一个名为 `emissions` 的空数据框。调用 `edit()` 函数，指定一个对象来保存空数据框，这样就打开了编辑器，命令如下：

```
> emissions <- edit(emissions)
```

请记住，`emissions` 是空的。通过调用前面命令中的 `emissions` 对象，告诉 R 用你输入的任何编辑过的数据覆盖空数据框。双击你想输入 / 编辑数据的单元格，输入数据。完成编辑后，在 OS X 系统中点击电子表格左上角或在 Windows 系统中点击右上角的“X”。*不要*

点击 Stop，OS X 系统中的 Stop 在编辑窗口，Windows 系统的 Stop 在屏幕的顶部。一旦点击 Stop，将丢失所做的全部修改。输入数据后，仔细检查以确保没有错误。如果发现错误，就双击想要修改的单元格，然后输入正确的数据。如果有必要的话，可以使用前面的命令再次回到编辑器来修改。为了以后不需要重新输入，就可以再次使用数据框，可使用下面的命令保存这个数据框：

```
> save (emissions,file="emiss.rda")
```

这行命令把 `emissions` 数据框写入工作目录中的 emiss.rda 文件。假设你仍在同一工作目录中，可以使用以下命令读取数据：

```
> load("emiss.rda")
```

1.8.3 从外部文件读取

你可能已经有一个最喜欢的输入数据的工具了，对许多人来说，这个工具可以是电子表格程序，也可以是文本编辑器。我喜欢 Mac 上的 Numbers，当然 Excel 或者其他电子表格也不错。通常的做法是，在电子表格程序中创建文件，并将其保存到工作目录。文件存在工作目录以后，就可以将其读取到 R 并进行分析。

练习1-3

多产的英国作曲家爱德华·埃尔加（1857—1934）有两个（也许是）最著名的作品：一个是 Pomp and Circumstance（《威风堂堂进行曲》），它是无数毕业典礼上播放的进行曲；另一个是交响乐 Enigma Variations（《谜语变奏曲》）。虽然完整的 Enigma Variations 是交响乐演出中受欢迎的部分，但非常优美的第 9 首 Nimrod 变奏曲通常单独演奏，不仅由管弦乐队，还由其他的合奏者（音乐团体）或独奏者演奏。

在演奏音乐作品之前，一定要问的一个最基本的问题是："节奏应该是什么样的？"换句话说："应该以多快的速度演奏？"虽然作曲家通常会给出指示，但是一些作品已经得到了广泛的解释，甚至在最德高望重的音乐家之中，解释也不一样。学习其他音乐家如何演奏这一作品，对策划个人表演是很有指导意义的。表 1-4 中呈现的 Nimrod 节拍数据来自很多表演记录，我是 2013 年 11 月 9 日在 YouTube 上找到的这些表演。

表1-4：不同团体演奏Nimrod的时间

表演者	媒介	时间	水准
Barenboim-Chicago SO	so	240	p
Solti-London Phil	so	204	p
Davis	so	270	p
Remembrance2009	cb	236	p
Belcher	org	254	p

（续）

表演者	媒介	时间	水准
Bish	org	232	p
ColdstrGuards	cb	239	p
Pallhuber-3 Lions BB	bb	257	a
Bernstein-BBC	so	315	p
Dudamel-SBolivarSym	so	239	p
John	org	252	p
Sunshine Brass	bb	186	a
Mahidol Sym Band	cb	173	a
Hills	org	240	p
Grimethorpe CollB	bb	200	p
Barbirolli_Halle O	so	200	p
Stokowski	so	244	p
Boult-London SO	so	211	p
Kindl-Marktoberdorfer BB	bb	238	a
Carter-Charlotte CB	cb	196	a
Cord-IndianaU	bb	188	a
Mack-SUNYFredonia CB	cb	160	a
U Akron CB	cb	193	a
Akron Youth Sym	so	188	a
BP-Ostwestfalen	cb	198	a
Santarsola-MoldovaPO	so	320	p
Klumpp_NWD PO	so	187	p
Burke-MancunianWinds	cb	257	a
US Army Field Band	cb	235	p
EE-Phonograph	so	186	p
Niemczyk-NWC O	so	169	a
Allentoff-Brockport SO	so	200	a

`Nimrod` 数据集有 32 行（实例 / 观测值）和 4 列（变量）。此数据将成为 R 的数据框。

`Nimrod`码本

除了最简单的数据集外，所有数据集都需要一个“码本”，它解释了变量的每个值的含义。`Nimrod` 数据集的码本如下：

`performer`（*表演者*）

若有，为指挥和乐团的名字。必须至少纳入一个可用的名字以供研究。

`medium`（媒介）

`bb` 铜管乐队（brass band）
`cb` 管乐队（concert band）
`org` 管风琴独奏（organ solo）
`so` 交响乐团（symphony orchestra）

`time`（时间）

从第一个音符到最后一个音符的表演时间，以秒为单位，去掉预告、调音、鼓掌等时间。可替代的测量节奏的办法，即假定自始至终节奏相同。

`level/proficiency level of the performers`（水平/表演者的水平）

`a` 业余（或学生）
`p` 专业

`time` 变量是一个*定量变量*（quantitative variable）；也就是说，它是总数的度量。算术中可以使用定量变量，所以可以计算 `time` 变量的总和或平均值。这些是 R 的数值向量，正如 1.5 节讨论的那样。这个数据集中的其他变量都是*分类变量*（categorical variable），即分成类的观测值。有些人将分类变量称为*定性*（qualitative）或*名义*（nominal）变量。这些是 R 的字符向量。由于 `bb`、`cb` 等不是数字，所以我们不能计算 `medium` 的平均值，计算甚至没有意义。但是，关于分类变量，有一些事情我们可以做，比如找出 `bb` 或 `cb` 的*频率*（frequency）。我们也可以使用分类变量的值来分组。例如，根据 `level` 变量的值把数据集分成很多部分，从而比较业余组和专业组的平均时间。

可以用下列方式中的任意一种来输入数据：

- 把数据输入到你最喜欢的电子表格程序中，并以 .csv 文件格式将电子表格保存（导出）到你的工作目录，命名为 Nimrod.Tempo.csv。R 可以读其他文件类型，但 .csv 格式文件最简单，也最不容易出错。然后打开 R 并输入如下命令：

  ```
  > Nimrod <- read.csv("Nimrod.Tempo.csv",header=TRUE)
  ```

  若要读取没有标题的文件，则使用 `header=FALSE`。

- 如果想读取 Excel 文件而不将其转换为 .csv 文件，有一个名为 `XLConnect` 的包可用。`Xlconnect` 可以执行很多其他任务，如编辑电子表格，以及把 R 数据写到 Excel 文件。如果你用的是旧版本的 R（3.0 之前的版本），那么无法使用这个包。下面的代码演示了当 `Nimrod` 数据被保存为 Excel 文件格式并以 Nimrod.xls 命名时，如何读取 `Nimrod` 数据：

  ```
  >install.packages ("XLConnect")
  >library (XLConnect)
  >Nimrod2 <-readWorksheetFromFile("Nimrod.xls",
   sheet = 1, header = TRUE)
  ```

如果命令的长度超过一行怎么办

如果需要执行的命令太长（如上例中的命令），以至于超出了控制台中的一行，只需继续输入，R 会自动将其余的文本放在下一行。输入完命令之前，不要按 Return 或 Enter 键。如果在输完命令前按 Return 键，R 不明白你的请求，可能会返回一个含糊的错误信息。

实际上，不需要在计算机上安装 Excel 来使用这个软件包。有许多数据集可以从政府机构和其他各种来源自由获取，你可以下载 Excel 格式的。关于这个主题的更多信息，请参见附录 E。通过 `Xlconnect`，可以复制这些文件并读取到 R，用于自己的数据分析。这个包可以读或写 .xls 或较新的 .xlsx 格式。访问 http://cran.r-project.org/web/packages/XLConnect/XLConnect.pdf，可以找到完整的文档。

- 使用文档编辑器或者文字处理器创建名为 Nimrod.Tempo.txt 的文本文件，使用空格作为值与值之间的分隔符。文件可以如下方式读取：

```
> Nimrod <-read.table("Nimrod.Tempo.txt", sep = "",
  header=TRUE)
```

如果前面讨论的把数据放到 R 中的方法都不适用于你所处的情况，请咨询 R 帮助文件中的“R 数据导入 / 导出”部分。在“R 帮助”中包含该文件，“R 帮助”是基础安装的一部分。使用上述任何一种方法读取数据到 R 之后，使用下列方法之一查看并确认是否起作用：

```
> Nimrod # 打印出整个数据集

> head(Nimrod) # 打印出前6行

> fix(Nimrod)  # 在编辑器中打开Nimrod数据
```

最后一种方式将打开编辑器（见图 1-2），让你在必要时检查数据或改变数据的值。

你也可以给出 R 命令，以多种方式分析数据，如下所示：

```
> mean(Nimrod$time)
[1] 222.0938
> table(Nimrod$medium)        # 获取每个medium的数量
  bb  cb org  so
  5   9   4  14
```

还可以创建一些酷酷的图表，在适当的时候我们会学到这些图表。

R Data Editor

performer	medium	time	level
Barenboim-ChicagoSO	so	240	p
Solti-London Phil	so	204	p
Davis	so	270	p
Rembrance2009	cb	236	p
Belcher	org	254	p
Bish	org	232	p
ColdstrGuards	cb	239	p
Pallhuber-3 Lions BB	bb	257	a
Bernstein-BBC	so	315	p
Dudamel-SBolivarSym	so	239	p
John	org	252	p
Sunshine Brass	bb	186	a
Mahidol Sym Band	cb	173	a
Hills	org	240	p
Grimethorpe CollB	bb	200	p
Barbirolli_Halle O	so	200	p
Stokowski	so	244	p
Boult-London SO	so	211	p
Kindl-Marktoberdorfer BB	bb	238	a
Carter-Charlotte CB	cb	196	a
Cord-IndianaU	bb	188	a
Mack-SUNYFredonia CB	cb	160	a
U Akron CB	cb	193	a
Akron Youth Sym	so	188	a
BP-Ostwestfalen	cb	198	a
Santarsola-MoldovaPO	so	320	p
Klumpp_NWD PO	so	187	p
Burke-MancunianWinds	cb	257	a

图 1-2：R 数据编辑器中的 Nimrod 数据。可以使用该编辑器查看数据和改变特定的值

你可以请求 R 以便获得关于 Nimrod 数据集的一些一般信息：

```
> summary(Nimrod)
                    performer medium       time       level
 Akron Youth Sym        : 1    bb : 5   Min.   :160.0  a:13
 Allentoff-Brockport SO: 1    cb : 9   1st Qu.:191.8  p:19
 Barbirolli_Halle O    : 1    org: 4   Median :221.5
 Barenboim-ChicagoSO   : 1    so :14   Mean   :222.1
 Belcher               : 1             3rd Qu.:241.0
 Bernstein-BBC         : 1             Max.  :320.0
 (Other)               :26
```

使用与保存 emissions 数据集一样的方式保存 Nimrod 数据框（当然要用不同的文件名），因为在后面的练习中需要重新检索并获取这个数据框。命令如下：

```
> save (Nimrod,file="Nimrod.rda")
```

使用 load() 命令可以重新加载，如下所示：

```
> load("Nimrod.rda")
```

关于如何读取和导入外部文件，详见附录 E。

1.9 获取脚本

到目前为止，我们已经输入了单行命令。大多数时候，运行良好。但是有时候你可能想要执行一系列命令，并重复执行整个命令序列。如果序列很长或要重复很多次，这个过程就会变得很乏味。幸运的是，R 可以使用脚本（script）。脚本是一系列命令，按你想要执行的顺序排列。你可以使用文本编辑器创建脚本并将其保存到文件；然后获取脚本（source the script），即读取脚本并执行保存的命令。

下面来看一下具体的过程。假设你在不断地更新 `Nimrod` 数据，时不时在 Excel 电子表格中添加一些新的观测数据，并且想用 R 做一些分析，看看最新数据的情况。接下来要用一系列命令来做分析，这需要事先安装两三个包。如果你不确定计算机上安装了哪些包，可以使用如下命令查找：

```
> installed.packages()
```

如果没有 `gmodels` 和 `XLConnect`，那么使用如下命令进行安装：

```
> install.packages("gmodels")
> install.packages("XLConnect")
```

现在，你可以使用如下的一系列命令来实现数据分析。请注意，当使用命令块时，通常不会在每行命令前加 R 提示符（>）[1]：

```
# 下面一组命令是一个脚本
library(gmodels) # 需要使用CrossTable命令
library (XLConnect) # 必须已经安装XLConnect
Nimrod2 <- readWorksheetFromFile("Nimrod.xls",sheet=1,
  header=TRUE)
attach(Nimrod2)
CrossTable(medium,level,
  prop.r=FALSE,
  prop.c=FALSE,
  prop.t=FALSE,
  prop.chisq=FALSE)
# 以上命令打印出一张表,其中每一个单元格都有计数,
# 但没有百分号
perf_time <- summary(time)    # 保存summary的输出
title  = "Summary of performance times:"
cat(title,"\n", "\n")        # 打印标题和两个空行
print(perf_time)             # 打印summary(time)的结果
detach(Nimrod2)
```

每次想查看结果时，用键盘输入这些相同的命令有点麻烦，所以建议你使用编辑器创建一个文件来保存上面的命令。R 提供了一个文本编辑器。在大多数 R 的版本中，可以从 R 控

注 1：余下的许多示例代码将会写成脚本，在每一行的开始没有提示符（>）。此外，为便于阅读，长命令（如这个例子中的 `CrossTable()` 命令）往往被分成多行来写。

制台左上角的“文件”（File）菜单中访问文本编辑器。选择“新建文档”（New Document）或“新建脚本”（New Script）来打开一个文本窗口，并输入命令。在工作目录中保存编辑好的脚本，本例中命名为 NimTotals.R。然后，使用以下命令执行文件中的所有命令：

```
> source("NimTotals.R")


   Cell Contents
|-------------------------|
|                       N |
|-------------------------|


Total Observations in Table:  32


             | level
      medium |          a |         p | Row Total |
----- -------|---------- -|-----------|-----------|
          bb |          4 |         1 |         5 |
-------------|------------|-----------|-----------|
          cb |          6 |         3 |         9 |
-------------|------------|-----------|-----------|
         org |          0 |         4 |         4 |
-------------|------------|-----------|-----------|
          so |          3 |        11 |        14 |
-------------|------ -----|-----------|-----------|
Column Total |         13 |        19 |        32 |
-------------|------------|-----------|-----------|


Summary of performance times:

   Min.  1st  Qu.   Median    Mean  3rd  Qu.    Max.
  160.0    191.8     221.5   222.1     241.0   320.0
```

脚本中的 CrossTable() 命令创建了一个列联表（contingency table）或交叉制表（cross tabulation）。表的顶部是标题行，包括变量 level 的值。左列给出了变量 medium 值的名称。标题下的第一行显示了“bb”（铜管乐队）的信息。有四个铜管乐队是“a”（业余团体），一个是“p”（专业人士）。右列和底部的行为各行或各列的总数。例如，Row Total 列显示各种铜管乐队共有五个。统计学家称这些总数为临界值（marginal value）或临界（marginal）。

表的下面是变量 time 的概要信息——表演时间。我们看到最短（minimum）时间为 160 秒，最长（maximum）时间为 320 秒。时间分布的中心有两个量：一个是均值（mean）或者平均值，即用所有数字相加之和除以数字的总个数；另一个是中位数（median），有一半的数字比它大，一半的数字比它小。最后，有第一个四分位数（first quartile）191.8（即高于四分之一的数字的点），以及第三个四分位数（third quartile）241（即高于四分之三的数字的点）。

你也可能会发现，编写一个包含 library() 和 setwd() 命令的脚本，可以方便地使用一条命令执行许多条这样的命令，否则你可能需要单独输入这些命令。如果你已经下载了几个经常使用的包，最好一次性加载所有的包，而不是记住何时需要哪个包。虽然每次启动 R 时发出 source() 命令不是特别不方便，但有些人更喜欢让 R 自动获得脚本。这是有可能的，但对于不同的平台，这个方法有点不同。在 OS X 操作系统中，打开屏幕顶部的 R 菜单并选择“首选项”（Preferences），可以指定工作目录，以便每次启动 R 时应用该工作目录。配置工作目录，以便通过拖放包含启动时将要获取的脚本文件来启动 R。在 Windows 中，需要找到 .Rprofile 或 Rprofile.site 文件并编辑它，使之包括在启动时要执行的命令。输入以下命令来看一个例子：

```
> ?Startup
```

1.10 用户自定义函数

当需要准确地重复一系列命令的时候，获取脚本是一个很好的工具。然而，有时你可能想重复执行一些程序，但不总是用相同的变量或相同的参数。如果你想使用上一节所创建的脚本，但不总用在同一文件中，那么可以编写自己的函数来选择要检索和分析的文件。

用户自定义函数的一般格式如下：

```
name <- function (argument1, argument2,...){
    commands
}
```

假设你把自己的函数命名为 update，让它获取一个 Excel 文件，每次使用函数时都要命名该文件。下面这段与前一节的脚本几乎是相同的代码，可以完成该操作。参数 fn 出现在函数语句以及 Nimrod2 的语句中，表明当 R 执行命令时，无论用户在函数调用中提供什么样的参数，都将在 Nimrod2 命令中被取代：

```
# 用户自定义函数,且命名为update
update <- function (fn){
library(gmodels)
library (XLConnect)
Nimrod2<-readWorksheetFromFile(fn,sheet=1,header=TRUE)
attach(Nimrod2)
CrossTable(medium,level,
  prop.r=FALSE,
  prop.c=FALSE,
  prop.t=FALSE,
  prop.chisq=FALSE)
 # 以上命令打印出一张表,其中每一个单元格都有计数,但没有百分号
perf_time = summary(time)  # 保存summary的输出
title = "Summary of performance times:"
cat(title,"\n", "\n")  # 打印标题和两空行
print(perf_time)
```

```
detach(Nimrod2)
}
```

为使用此函数，首先必须如保存任意 R 脚本那样将其保存，然后加载或获取它。在准备按如下方式调用函数之前，可以发出任意多的命令，其中 myfile.xls 是你要分析的 Excel 文件的名字：

```
>update("myfile.xls")  # 文件名在引号中是因为myfile.xls是字符型变量的值
```

当然，下一次你可能想要分析不同的文件，只需将文件名替换为新文件的名字即可。你还可以创建简单的数学函数或相当复杂的程序，例如用来生成一种特殊类型的图，后面我们会看到这些图。

1.11 开始令人享受的事

图 1-3 展示了几张基于 `Nimrod` 数据集生成的图。本书后面几章将讨论所有这些类型的图，当你读完本书，应该能够生成任意一种或更多的图。

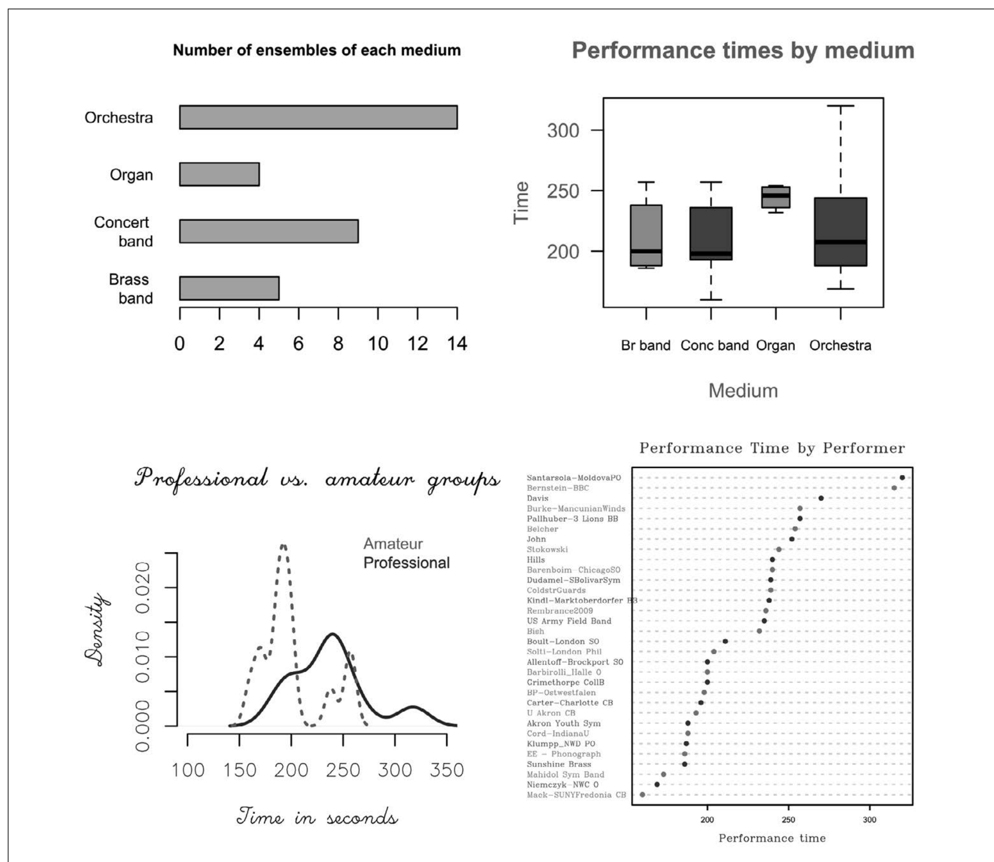


图 1-3：基于 `Nimrod` 数据集生成的几种类型的图

练习1-4

在练习 1-2 或者练习 1-3 中，如果输入数据时遇到问题，按照如下两种方法，试着输入练习 1-1 中简单的数据集。首先，数据如下：

存活率	使用年数
1	94
2	92
3	90
4	80

方法1：电子表格

打开电子表格，输入 5 行（包括标题）2 列的数据，如下所示。把文件以 .csv 格式导出到工作目录（详见 1.7 节），并命名为 simple1.csv。然后，创建新的数据框 `mydata`。代码如下：

```
> mydata = read.csv("simple1.csv",header=TRUE)
> mydata

  year rate
1    1   94
2    2   92
3    3   90
4    4   80
5   NA   NA
```

本例的电子表格中有一行空白，因此R数据框的最后一行存在缺失值。可以使用参数 `nrows`，只读入指定的几行数据来处理缺失值。代码如下：

```
> mydata= read.csv("simple1.csv",header=TRUE,nrows=4)
> mydata

  year rate
1    1   94
2    2   92
3    3   90
4    4   80
```

如果在标题前有其他（空白）行，结果可能如下所示：

```
> mydata

   X  X.1  X.2 X.3
1 NA year rate  NA
2 NA    1   94  NA
3 NA    2   92  NA
4 NA    3   90  NA
5 NA    4   80  NA
```

可以使用参数 `skip` 忽略第一行，代码如下：

```
>mydata= read.csv("simple1a.csv",header=TRUE,skip=1, nrows=4)
> mydata

   X year rate X.1
1 NA    1   94  NA
2 NA    2   92  NA
3 NA    3   90  NA
4 NA    4   80  NA
```

在最后一个例子中，有额外的列。这不是一个大问题，忽略即可。重要的是两个向量 `year` 和 `rate` 有正确的行数。如果想删除其中一个无用的列，可按如下方式处理：

```
>mydata$X = NULL
>mydata

  year rate X.1
1    1   94  NA
2    2   92  NA
3    3   90  NA
4    4   80  NA
```

方法2：文本

打开文字处理器或者文本编辑器。输入数据，每一行中输入的两个数字之间用空格分开，并在每行结尾回车，如下所示：

```
year rate
1    94
2    92
3    90
4    80
```

多余的空格无关紧要，但是数据应该被保存为纯文本，而不是富文本。如果你的文字处理器允许保存为 .txt 文件，使用“另存为”命令将文件保存到工作目录，并命名为 simple2.txt。否则，需要使用导出命令导出文件，同样命名为 simple2.txt。将数据读入 R，并使用如下命令创建新数据框 `newdata`：

```
> newdata = read.table("simple2.txt",sep="",header=TRUE)
> newdata

  year rate
1    1   94
2    2   92
3    3   90
4    4   80
```

第 2 章

R图概述

本章讨论怎样导出图表以及探索性图表和展示性图表间的区别，也会简单概述 R 中的几种图形系统。如果你有编程经验或者丰富的图表经验，在了解 R 图表类型的特性前，很可能会想先了解这些内容。如果你没有这样的背景，可能会觉得这一章技术性有点过强，此时此刻没必要学习。若是这种情况，则可直接转到第 3 章，准备好后再来看这一章。

2.1 图表导出

制作一张图表后，你可能想保存它或者将它存入文件。怎样保存取决于你正在使用的软件是什么。例如，如果使用文字处理软件，可以简单地复制图表，方法是打开 R 的图形窗口，点击“编辑”（Edit）菜单中的“复制”（Copy），或者点击链接到图表的上下文菜单，然后就可以把图表粘贴到文字处理软件。

其他软件需要你付出更多的努力。若尝试了复制粘贴的方法但不起作用，就需要选择一个文件类型，并使 R 以这种格式保存图表到指定文件。图片格式有 .bmp、.pdf、.jpeg、.png、.tiff、.ps（PostScript），等等。可以以这些格式中的任意一种保存图片。下面的代码示例展示了怎样将练习 1-1 中制作的图表保存为 .jpeg 文件，命名为 test.jpeg（当然，可以用任意其他名字来命名，只要扩展名是 .jpeg。比如，将它命名为 mywork.jpeg）：

```
jpeg("test.jpeg")  # 打开图案
plot(year, rate)
dev.off()  # 关闭图案,必须执行这条
```

图表按此方式保存以后，可以插入文字处理程序的文档。比如在 OpenOffice 中，可以打开“插入”（Insert）窗口，点击“图片”（Picture），再点击“从文件”（From File），然后从工作目录中选择 test.jpeg 文件。当然，当图片保存在一个文件中后，可被加载到各种应用程序，比如绘图或插图程序。例如，如果需要，可以在 Adobe Illustrator 或 Inkscape 中“美化”（brush up）你的 R 图。本书中的图表不“美化”，以 .png 格式保存在我的编辑的 Google Drive 账号中。对于高分辨率图，我使用如下这段代码：

```
dpi=600
png("filename.png", width = 6*dpi, height = 6*dpi,
  res = dpi)
graphic commands
dev.off()
```

要了解更多这种保存文件的方式，可输入 **?png** 进行查看。

什么是“设备”

指定设备是告诉 R 在哪里画图，并定义 R 将采取的格式。如果没有指定设备，图形出现在计算机屏幕上的图形窗口中。如果想将图表保存到文件（在硬盘、闪存或者其他地方），必须通过“打开设备”（opening a device），告诉 R 将图表写入哪个文件（设备）。通过命令告诉 R 使用的设备以及 R 文件的格式，比如 .jpeg、.pdf、.png 等。然后，可以使用任何需要的图形命令在设备上画你想要的类型的图。最后，“关闭设备”（close the device），即停止写入该文件。如果已经（使用 `dev.off()`）关掉了设备，则结果会显示在屏幕上，或者写入一个打开的新设备。输入 **?device** 命令，可了解更多关于设备的信息。

2.2 探索性图表和展示性图表

图表有利于探索（exploration）和展示（presentation）。探索是分析数据和寻找关系与模式的过程；展示成果是将你的发现展示给其他人，他们没有像你一样深入研究过这些数据。在探索数据期间，你的图表可能是粗陋的、简单的，某种程度上没有吸引力。作为数据分析师，你了解数据，并随着每张图表的制作，了解越来越深入。有人可能期望看到所有的标题、标签、参考细节和颜色，他们发现这些是必要的，但这些对你并不是必需的。另外，添加所有无关紧要的细节只会减慢探索的速度。同时，一些图表会被证明是行不通的或不是很有趣。因此，在探索的过程中，很多图表可能会被丢弃。

随着探索过程的继续，添加一些细节可以让关系更加清晰。越接近展示或发布，图表就会越详细、越漂亮。在分析数据的过程中，你可能会创建很多单调的图表，最终报告中的漂亮图表则相当少。

下面是 mtcars 数据集的两张图表，R 的基础安装中包括该数据集。它展示了 mpg（英里 / 加仑）和 wt（汽车的重量）间的关系。第一张图（见图 2-1）是用散点图（scatter plot）辨别这两个变量间关系的早期尝试。它清楚地表明，随着汽车重量的增加，每加仑行驶的里程减少。如果你不熟悉散点图，在读完第 12 章后，可能会想回到这个例子。第二张图，如图 2-2 所示，比第一张图精致。它有标题和坐标轴上的标签，并按气缸数将汽车分类，当然图片还是彩色的。这是可以在演示文稿（PowerPoint）上展示的图表。在这两个例子之间，可能有其他数个相对单调的探索性图表。因为本书主要介绍图表分析的过程，所以书中的很多例子简单、朴实无华，但它们都会通向有吸引力的成品。

生成图 2-1 的一行代码如下：

```
plot(mtcars$wt, mtcars$mpg, pch=16)
```

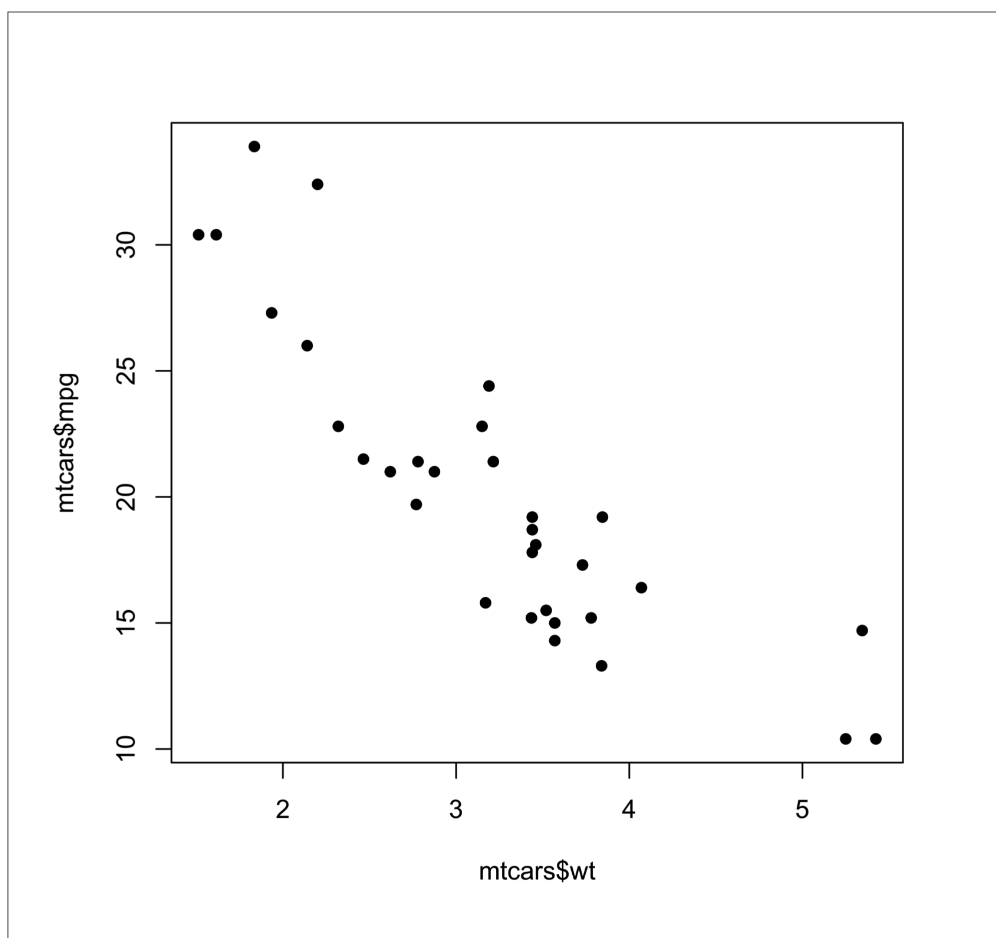


图 2-1：wt 和 mpg 的探索性图表

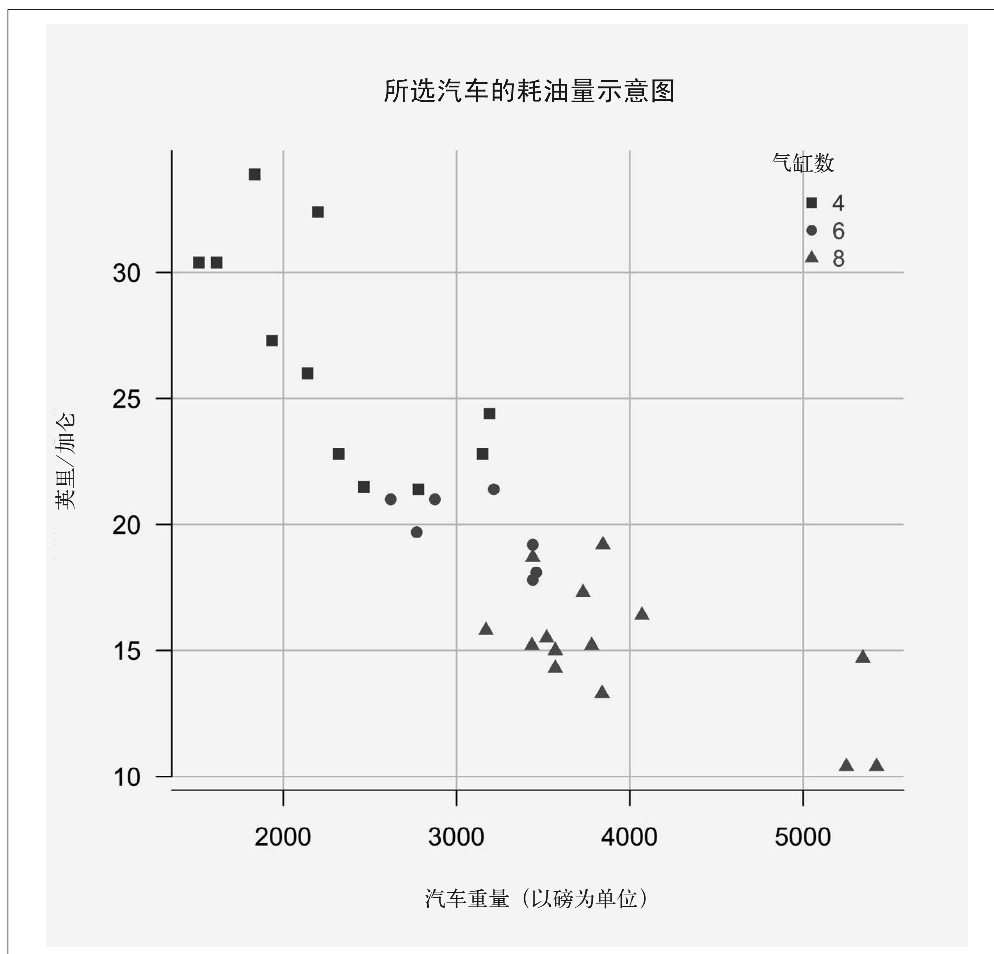


图 2-2：wt 和 mpg 的展示性图表；图 2-1 的改进版

生成更丰富多彩且更精致的图 2-2 需要多几行代码。完成它需要做更多的工作，但是它可以作为有用的展示对象，付出的努力也是值得的。生成图表使用的各种类型的命令在此不做介绍，本书后几章会对它们进行讨论。要点就是用 R 很容易制作出简单有效的图表，但是若要制作精美的图表，则需要多做一些工作。生成图 2-2 的脚本如下：

```
# 生成图2-2的脚本
library(car)
attach(mtcars)
par(bg="snow",fg="snow",col.axis="black",bty="l")
mtcars$wt2 = 1000*wt
attach(mtcars)
scatter plot(mpg~wt2|cyl,
  smoother=FALSE,
  reg.line=FALSE,
  col=c("indianred4","blue","purple"),
```

```
  pch=c(15,16,17),
  main="Fuel Consumption in Selected Cars",
  ylab="Miles per Gallon",
  xlab="Weight of Car in Pounds",las=1,
  legend.plot=FALSE,bty="l")
axis(2,col="black",at=c(10,15,20,25,30,35),las=2)
axis(1,col="black",at=c(1000,2000,3000,4000,5000,6000))
legend("topright",
  title="No. of Cylinders",
  c("4","6","8"),
  inset=-.005,
  text.col=c("indianred4",
  "blue","purple"),
  title.col="black",
  cex =.65,
  pch=c(15,16,17),
  col=c("indianred4","blue","purple"),
  bty="n")
detach(mtcars)
```

2.3 R图形系统

R 有几种可用的图形系统。基础 R 包有很多好用的图表函数，但 R 用户也通过贡献新的图形软件包扩展了 R 的制图能力。下面的讨论描述了各种图形包的优势和风格。

2.3.1 基本图形和网格

基础 R 包括图形软件包，在第一次安装 R 的时候会自动安装，且每次启动 R 的时候也会自动加载。它可以生成广泛定制的多种类型的图表，功能非常强大。很多 R 用户永远不需要比基础 R 提供的更强大、更灵活的功能。本书的大多数图表是用基础的 R 图形软件包生成的。

尽管基础 R 图形包让人印象深刻，但有时应用程序要求对图形输出的细节有更多控制。因此，针对低级（low-level）的图形，开发了网格（`grid`）包。"低级"是指 `grid` 提供了大量的工具或材料，供使用其他包的开发者利用，来完成图表。

在这方面，`grid` 有点像一个生产木板（低级材料）的木材厂，建筑商或家庭业主将这些材料用在房屋内的（高级）项目，比如地板或者书架子上。好的建造者不需要关心木材厂如何截断树木、粗切木板及刨光。建造者的工作从木板开始，而不是从树木开始。`grid` 包加工材料，这些原料可用于本章所讨论的其他图形系统，以及其他多种 R 包中的一些图形处理程序。`grid` 包不提供任何能直接用于完成图表的函数。我们需要用到的一些图形函数已经使用 `grid` 函数构建完成。关于 `grid` 的详细信息，参见 Murrell（2011）。因为用户通常不直接编写网格代码，所以这里没有与 `grid` 相关的例子。

2.3.2 lattice

开发 `lattice` 包是为了针对*多元数据*（multivariate data）提供改进的图表，即一次绘制

超过两个变量的图形。lattice 仿照了 Cleveland（1985，1993）描述的格子图（trellis graphics）。他认为，有时可视化几个变量间关系的最有效的方法，并不是试图把所有变量放在一张图表中，而是查看几个按目的组织的相关的图表。例如，图 2-3 展示了一个来自 epicalc 包的 BP 数据集格子图，有四个窗口或面板（panel）。在每个面板中有一张收缩压 / 舒张压的图，每个面板展示了 sex（性别）和 saltadd（是否在饮食中加盐）的组合图。

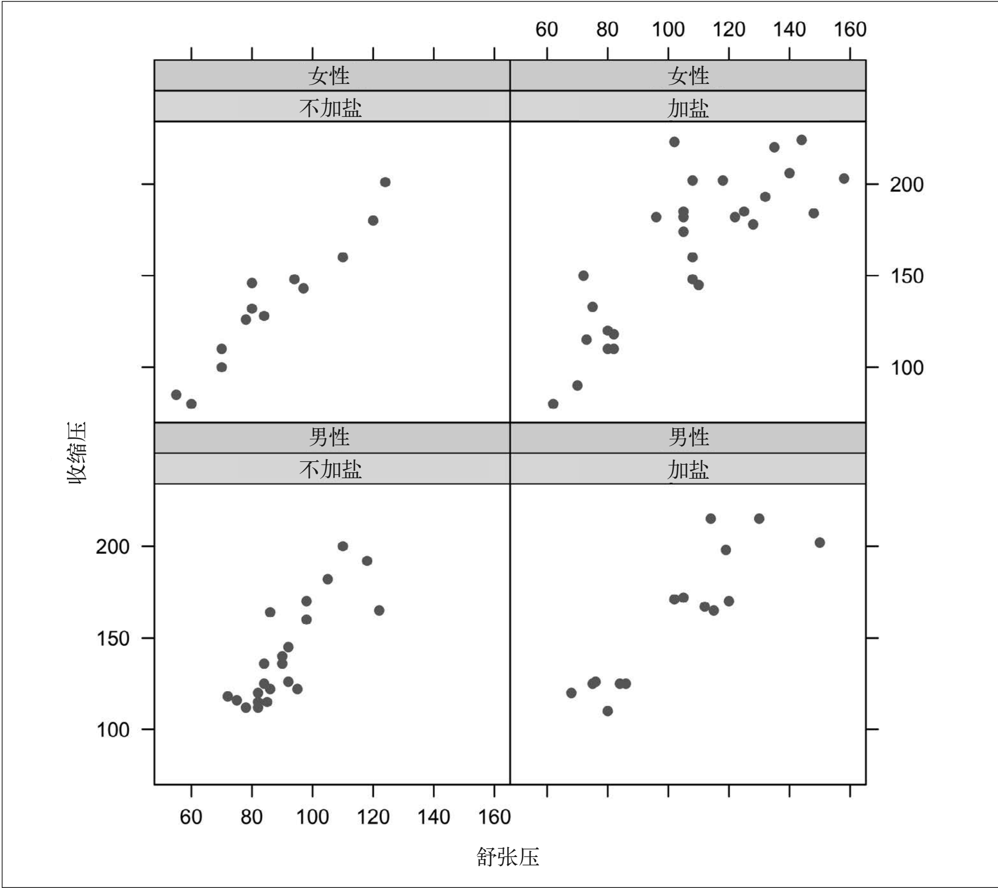


图 2-3：使用 lattice 包生成的格子图

通过在一个页面上浏览四个相关的图表，图 2-3 展示了一种同时研讨四个变量间关系的方式。代码如下：

```
# 图2-3
library(lattice)
library(epicalc)
attach(BP)
xyplot(sbp~dbp|saltadd*sex,pch=16)
detach(BP)
```

lattice 在安装基础 R 时会自动安装，但在需要它的每个会话中必须加载它。除了格子图，它还包括许多其他类图表函数。虽然本书只使用了几个 lattice 的例子，但它是一个很好的 R 图形包，扩展了 R 的功能。当你越来越熟悉 R 和基础图形时，会发现它值得花时间学习。

2.3.3 ggplot2

设计 ggplot2 包是为了在访问所有的图表类型时语法一致；也就是说，从一种图形到另一种图形，命令语言是相似的。这和基础 R 形成鲜明对比，因为它有许多可用于多个不同类型的图表参数，但也有一些差异。ggplot2 包功能也很强大，你可以很轻松地定制图表的显示。因为这个包的语法和基础 R 差异很大，所以本书很少有使用它的例子。不过，应该提一下，有些命令的设计看起来和基础 R 很类似，所以可以尝试一下 ggplot2 的功能，不需要付出多少努力。如果你需要这个包的一些特殊功能，在更好地理解 R 以后，可以学习一下。ggplot2 和基础 R 图形的审美风格相当不同，你可能喜欢也可能不喜欢它。图 2-4 是它的一个例子，由如下代码生成：

```
# 图2-4
library(ggplot2)
ggplot(mtcars, aes(x=wt, y=mpg)) + geom_point()
```

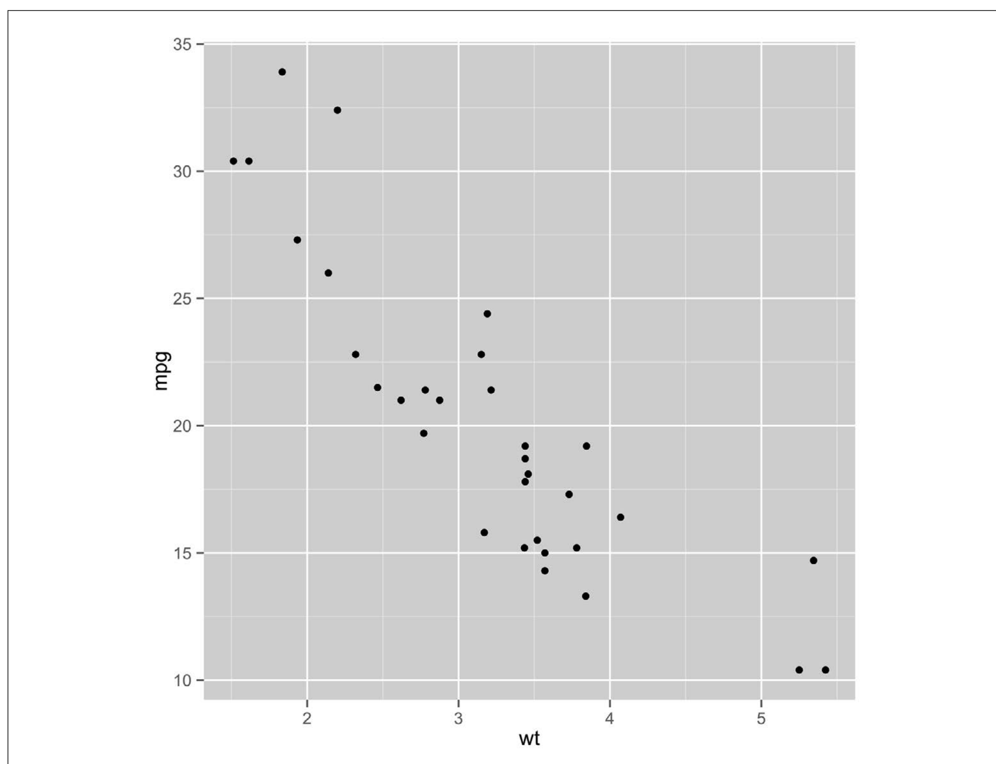


图 2-4：基于与图 2-1 和图 2-2 中基础 R 图相同的数据，用 ggplot2 生成的简单图表

`ggplot2` 不是随基础 R 包一起安装的，因此，如果需要使用它，首先要安装它，然后在想使用它的每个会话中加载它。

2.3.4 包的特殊应用程序/图表

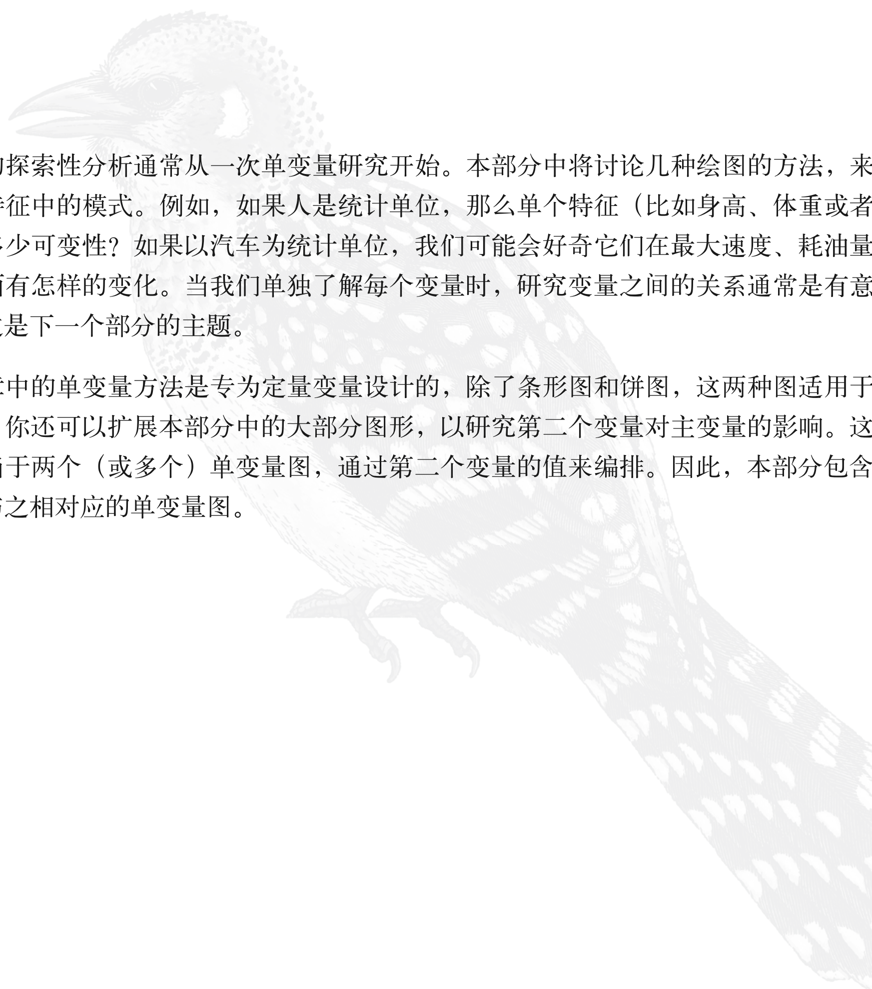
许多包内含一些绘图功能，即便那些不是主要用作图形包的包也是如此。在 CRAN 任务视图的 Web 页面（http://cran.r-project.org/web/views/），你可以领略到丰富多样的图形产品。点击 Graphics 查看许多包中的图形类型的概述，但请记住，这并不包括所有的图形类型。此外，一些包可能只有几个混有许多其他特性的图表函数，而这些类型的包通常不会出现在任务视图中。用你最喜欢的搜索引擎在互联网上搜索 R 中某种图形系统的相关信息。在成千上万个 R 包中找到你想要的包是一个艰巨的任务！

2.3.5 用户自定义图表函数

如果找不到适合数据的图表，可以自己写一个图表函数。这只是第 1 章提到的方法的扩展。后面将介绍大量的图形工具，你可以将它们用到这些函数中。第 14 章提供了用户编写的用来生成 Bland-Altman 图的图形函数的例子。

第二部分

单变量图



对数据集的探索性分析通常从一次单变量研究开始。本部分中将讨论几种绘图的方法，来发现单一特征中的模式。例如，如果人是统计单位，那么单个特征（比如身高、体重或者智商）有多少可变性？如果以汽车为统计单位，我们可能会好奇它们在最大速度、耗油量或马力方面有怎样的变化。当我们单独了解每个变量时，研究变量之间的关系通常是有意义的，但这是下一个部分的主题。

接下来几章中的单变量方法是专为定量变量设计的，除了条形图和饼图，这两种图适用于分类变量。你还可以扩展本部分中的大部分图形，以研究第二个变量对主变量的影响。这些扩展相当于两个（或多个）单变量图，通过第二个变量的值来编排。因此，本部分包含扩展图和与之相对应的单变量图。

第3章 带状图

3.1 一种简单的图

带状图（strip chart）是最简单但仍然非常有用的一种图。一些分析师称其为“点图”（dot plot）。此类图提供了一种查看一组数字如何分布（distribute）的方法。也就是说，数据的形状是什么？可以确定最大和最小数字吗？它们是如何展开的，以及一些数字是否聚集在一起？

我们来研讨一下基础 R 包提供的 trees 数据集。为了查看该数据集的描述，请输入如下命令：

```
> ?trees
```

该命令会打开一个新窗口，描述了该数据集。窗口中提供的信息如下：

简介

  这是关于数据的简单概述。

使用

  显示数据集的名字。

格式

  说明数据集的结构是数据框，有 3 个变量及 31 个观测值，并给出变量名和测量单位。

来源

  标明数据来自哪里。

参考文献

包含数据分析例子的书籍或文章。我有时复制并粘贴参考文献到搜索引擎，来查看其中的分析示例。这种方法并不总是成功的，但一旦可行，通常是非常有用的。

例子

提供一些使用数据集的 R 代码。复制并粘贴一个或多个例子到 R 控制台，看看生成了哪类统计分析或图形，有时这么做很有趣。本书稍后将讨论生成的图形类型，但也有一些统计分析没有涉及。

大多数数据集的帮助文档提供了上面列出的多项信息。

> **当你需要一点帮助……**
>
> R 提供了几种易于访问的帮助。举例如下。
>
> - `?numbers` 为以 `numbers` 命名的数据集提供帮助。
> - `?mighty` 为以 `mighty` 命名的函数提供帮助。
> - `example(`*x*`)` 提供函数 *x* 的样例输出。请尝试命令 `example(stripchart)`。
> - `vignette()` 列出安装在计算机上的所有包的简介。试试看！
> - `vignette(`*x*`)` 展示 *x* 的简介。简介几乎可以是任何内容，从用户手册到 R 代码以及示例输出。

`trees` 数据集中的所有变量以不同的方式测量样本中 31 棵树的大小。考虑第一个变量 `Volume`，试试下面的命令。确保 `Volume` 中的字母 V 是大写的；否则 R 将报错，因为它不识别变量 `volume`。

```
> attach(trees)
> Volume
```

你将会看到如下结果：

```
 [1] 10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 24.2 ...
[19] 25.7 24.9 34.5 31.7 36.3 38.3 42.6 55.4 55.7 58.3 51.5 ...
```

R 打印出 31 棵树各自的体积。输出包含索引号，显示第一行始于 `[1]`，`[1]` 为向量的第一个元素。第二行始于第 19 个元素。处理这些信息可能需要一段时间，因此需要一点策略。也许首先应该寻找最小值和最大值，也可以试着猜测平均值，或者看看是否有些树的体积相同。即使像这样的一个相对较小的数据集，这个过程看起来都似乎很难。下面的代码尝试一个简单的图：

```
> stripchart(Volume)
```

带状图如图 3-1 所示。

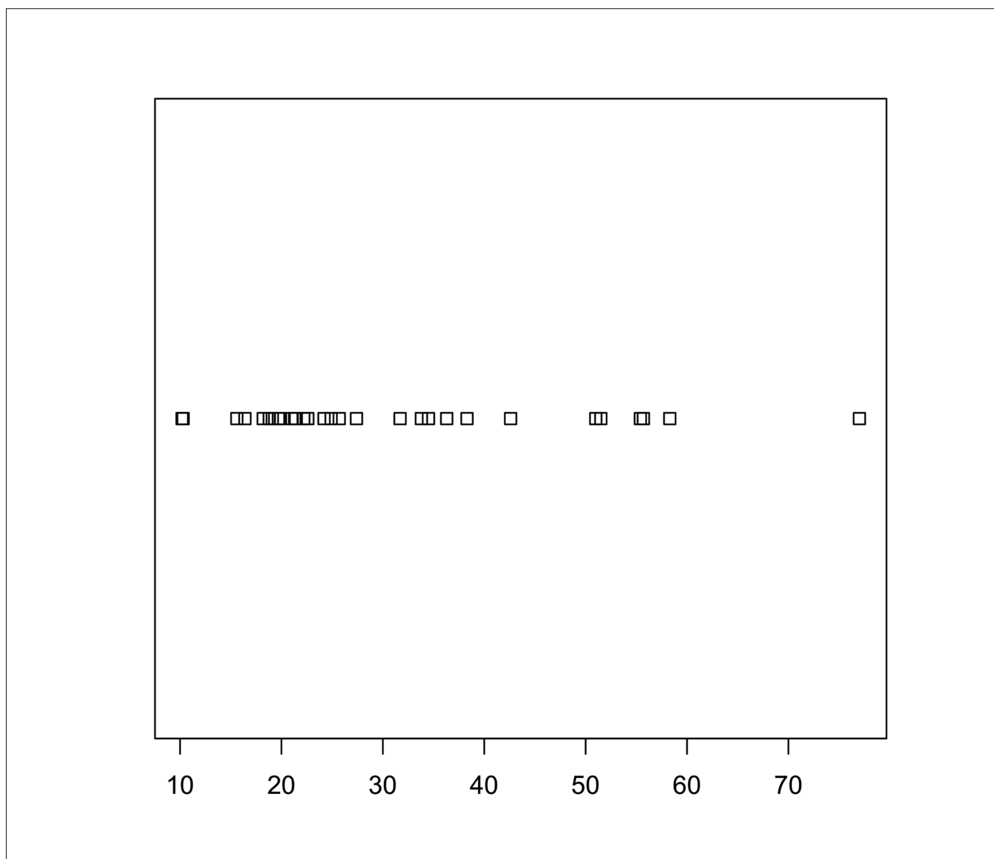


图 3-1：变量 Volume 的带状图

底部坐标轴显示 31 棵树的体积的数值。图形显示许多值集中在 20~30 立方英尺附近，少量的值集中于 50~60 立方英尺。有一个非常大的值，远远超过 70。这么大的值会引出一些重要的问题。是否有一些可以解释异常值的因素被忽视了？是否在测量或记录树的测量值时出现了错误？是否有一些方法来验证或纠正异常数据？如果要进一步分析这些数据，这个异常值应该包含在内还是排除在外？所有这些可能被忽视的问题，使分析从只考虑数据开始——换句话说，仅考虑平均体积。

图中没有 31 个不同的方块，因为有些树的体积相同或接近。通过在 stripchart() 命令中添加一个参数，可以把在图中争夺相同位置的那些树分散开，如下所示：

```
> stripchart(Volume, method = "jitter")
```

图 3-2 比图 3-1 有明显的改善。

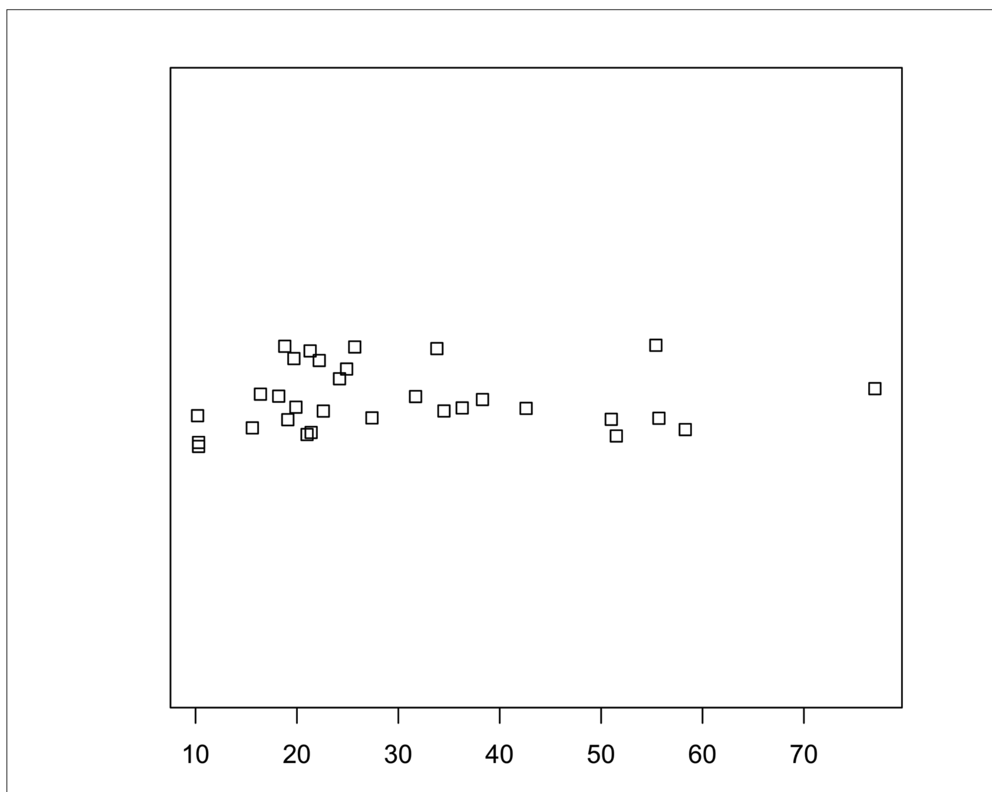


图 3-2：树体积的抖动带状图

图 3-2 中仍有一些重叠，但仔细数数现在有 31 个点。共享同一位置的一些点“抖动”着高于或低于彼此，使图更易于阅读。进一步改善该图的一个方法是增加图中抖动量。可以使用帮助命令查看可用选项，命令如下：

```
> ?stripchart
```

`stripchart()` 的帮助文件有很多有用的信息，包括很多本书没有介绍但也许对你很有帮助的信息。帮助文件包括以下几个部分。

描述（Description）
: 表明函数的作用，以及适用于哪些情况。

使用（Usage）
: 说明基本的语法和默认选项。换句话说，如果不指定特定参数，将会发生什么？

参数（Argument）
: 描述了可以做的选择，通过选择来达到自己想要的结果。（参数是在 1.2 节中定义的。）

细节（Detail）

　此处提供各种细节，比如去哪里找到更多信息、函数是谁开发的，等等。

举例（Example）

　提供代码，你可以将其复制粘贴到控制台，来感受下函数的真正功能。

帮助文件显示，`jitter` 的默认值是 `0.1`。用参数 `jitter` 和参数 `stack` 做实验，看看它们是如何工作的。优化图形的另一种方法是使用参数 `pch`（可看作 plot character）改变代表每个点的符号。这一次，使用参数 `xlab` 为 *x* 轴添加一个标签，代码如下：

```
> stripchart(Volume, method = "jitter", pch = 20,
  xlab = "Volume in cubic feet")
```

改变后的结果如图 3-3 所示。

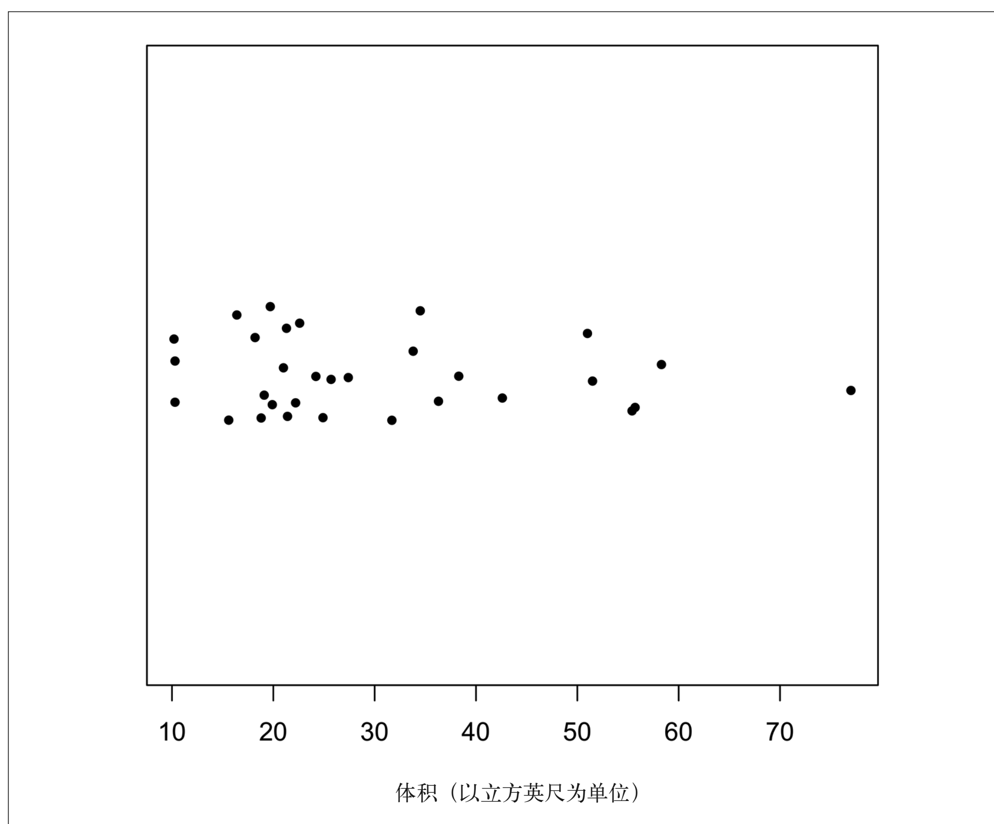


图 3-3：增加抖动并改变打印符号的体积带状图

因为打印符号较小，并且不像之前的图那样出现重叠，所以图 3-3 中各树的体积的间隔非常清晰。若仔细看图 3-2 和图 3-3，你会发现点的位置似乎有点不同了。实际上，它们都

是对的！当使用 `method = "jitter"` 时，点是随机（random）放置的，高于或低于地平线。因此，即使是完全相同的数据，每次执行带抖动的 `stripchart()` 命令，结果也会略有不同。垂直方向（vertical placement）上的位置会略有不同，但水平位置则完全相同，而水平位置才是我们所关心的。（实际上，有一种方法可以解决这个问题，但会带来一个技术问题，这里我不想介绍。如果想看每个图中在同一个位置的点，那么每次使用 `stripchart()` 命令时，在其前边输入 `set.seed(1)` 命令。）

看下面的命令：

```
> ?points
```

它将打开一个窗口，窗口中包括参数 `pch` 的选项，还有很多其他信息。图 3-4 提取出了 `pch` 选项，并以更方便的形式展示出来。

pch 的值

0 □	5 ◇	10 ⊕	15 ■	20 •	25 ▽
1 ○	6 ▽	11 ✡	16 ●	21 ○	
2 △	7 ⊠	12 ⊞	17 ▲	22 □	
3 +	8 ✳	13 ⊗	18 ◆	23 ◇	
4 ×	9 ✜	14 ⊠	19 ●	24 △	

图 3-4：图形参数 pch 的选项

对带状图来说，最好的符号是那些重叠最少的符号，往往是空心圆（`pch=1`）或非常小的符号（`pch=20` 或 `pch=18`），还可以使用其他字符。因此，如果想要非常小的图形标记，可以使用英文句号：

```
pch = "."
```

3.2　数据可以漂亮

图 3-3 是一个非常好的展示数据的方式，便于理解数据。然而，为了达到展示的目的，我们可能更喜欢引人注目的东西。为使这张图更加有趣一点，可以做几件事。此处讨论的内容对于后文中的图形也是有帮助的。

大多数 R 图的四周都有框。这些框通常不是必要的，看起来缺乏吸引力，而且也会分散注意力。（有些人喜欢有框，但和我一起学习，你可能会在某个时刻想知道如何去掉框。）幸运的是，可以很容易地将它们去掉。有许多参数可以用来控制图形输出。查看图形参数的列表，请输入以下命令：

```
> ?par
```

可以在不同的图形命令中使用这些参数。其中一些参数只能在 par() 命令中使用，par() 命令通常出现在调用另一个图形函数之前。控制图框的参数是 bty，即框的类型（box type）。在输入图形命令之前，发出 par(bty="n") 命令，以完全去掉框。以下两条命令生成没有框围绕的图 3-5：

```
> par(bty = "n")  # 图3-5
> stripchart(Volume, method = "jitter", pch = 20,
  xlab = "Volume in cubic feet")
```

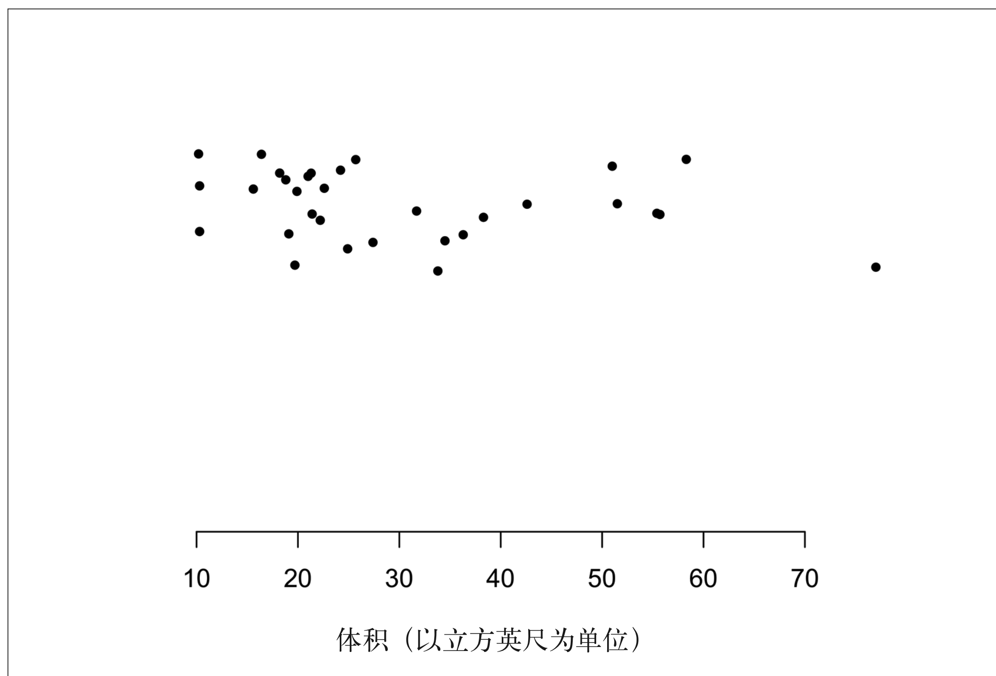


图 3-5：体积的带状图，周围没有框

图的周围有框吗

图的周围可以四周有框、局部有框或都没有框。par() 函数的参数 bty 可用的选项有 "o"（默认）、"l"、"7"、"c"、"u" 或 "]"。每一个参数可在图周围，以参数表示的形状创建一个局部框。bty = "n" 表示去掉整个框。理解这些的最好方法是亲自尝试使用其中一个或多个。欲了解更多信息，请输入 **?par** 查询。

注意，图 3-5 中的图没有框，Volume 的单个极值似乎没处于适当的位置，超出了图像范围。可以通过使用参数 xlim 延长坐标轴来解决这个问题，如图 3-6 所示。

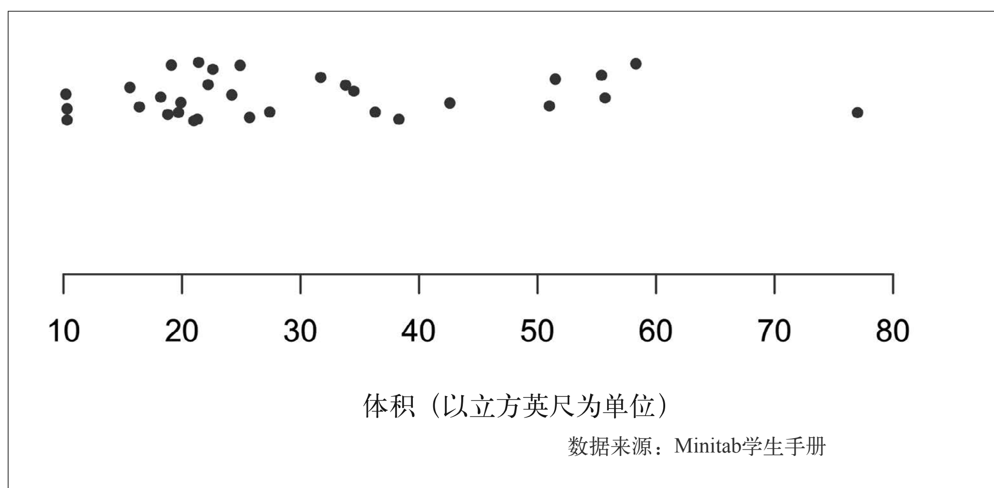


图 3-6：没有框、添加颜色、延长坐标轴并添加文本的带状图

接下来想到的可能是颜色。在 stripchart() 命令中再添加一个参数，很容易就能改变点的颜色，代码如下：

```
# 图3-6
par(bty = "n")
stripchart(Volume,
  method = "jitter", jitter = .3,
  pch = 20,
  xlab = "Volume in cubic feet",
  col = "dodgerblue4",
  ylim = c(0,8),
  xlim = c(10,80)) # xlim 强制 x 取值位于 10 ~ 80
  # 且ylim更接近x轴
```

注意，参数 col 给出颜色的名称。R 有 657 种已命名的颜色可供使用。可以在附录 B 查看颜色列表。通过如下命令，也可以在你的计算机屏幕上看到 R 的颜色：

```
> demo(colors)
```

前面命令生成的图，只是改变了图中点的颜色。看完该图之后，你或许也想改变坐标轴的颜色。使用 axis() 命令可以改变坐标轴的颜色，代码如下。直接操作现有的图。axis() 命令改变了前面命令生成的图的坐标轴的颜色（对比图 3-6，再次查看结果）：

```
> axis(1, col = "dodgerblue4", at = c(10,20,30,40,50,60,70,80))
```

坐标轴的控制

axis() 函数为当前平面图增加坐标轴。axis() 命令中使用的参数如下。

- 操作哪个坐标轴：1。该参数是必需的，并且不需要包含参数名称（side）。可能的值为 1 = 下、2 = 左、3 = 上和 4 = 右。
- 坐标轴的颜色：col = "dodgerblue4"。
- 标记和标签的位置：at = c(10,20,30,40,50,60,70,80)。

还有其他几个参数，允许指定图的特性，比如字体、标签、线宽等。该函数可以添加到大多数图表，不仅仅是带状图。axis() 不是每个图必需的函数，但它使你对最终结果有很大的控制。欲知更多信息，请输入 **?axis** 查询。

mtext() 命令在图形的绘图区域这外增加一个注解：

```
> mtext("Data source: Minitab Student Handbook",
side = 1, line = 4, adj = 1, col = "dodgerblue4", cex = .7)
```

给图添加文本

可以使用 text() 和 mtext() 函数在图的空白区域或页边距处分别添加文本。对带状图例子中的 mtext() 命令的说明如下。

- 需要显示的文本：本例中为 "Data source: Minitab Student Handbook"。
- 位置：side = 1 指定在图形下方。 其他可能的值有 2（左）、3（上）和 4（右）。
- 图形下方（或者上方、左边、右边）线的数量：line = 4。
- 对齐：adj = 1 表示在最右边，adj = 0 表示在最左边，0~1 的值表示与左边和右边的距离。
- 颜色：col = "dodgerblue4"。
- 类型大小：cex = .7。

欲知更多信息，请输入 **?mtext** 或 **?text** 查询。

结束 trees 数据集，输入如下命令：

```
> detach(trees)
```

3.2.1　练习3-1

尝试用另一个数据集 mtcars 绘制带状图。为变量 mpg 制作一个简单的带状图。你学到了什么？观察按汽车拥有的气缸数分解的 mpg 带状图，尝试制作一个更复杂的图；也就是说，为每个 cyl（气缸数）值绘制一个 mpg 带状图。下面的命令通过在符号 ~（波浪号）之后，使用分组变量 cyl 实现图形。通常分组变量是一个分类变量，而在本例中，它实际上是一个数量变量，只有少量可能的值——汽车可能有 4、6 或 8 个气缸：

```
> attach(mtcars)
> stripchart(mpg~cyl)
> detach(mtcars)
```

这是一种改进吗？你从这个图中获取了哪些额外的信息？试着给这个图增加抖动。这会有帮助吗？

3.2.2　练习3-2

很多可以添加到 R 的包为基础 R 提供的函数提供替代方案。通过安装 plotrix 包，尝试用 dotplot.mtb() 函数替换 stripchart()，测试 trees 数据集的变量 Volume。后者与 stripchart() 相似的地方是什么？有什么不同吗？你更喜欢哪个函数呢？

第4章

点图

基础点图

点图（dot chart，有时也称为 dot plot）和带状图非常相似，它们都展示点如何分散或聚集。点图还让我们能从数据中收集更多的信息。也许你认为接下来的数据集有点可怕，但想一想，本书的一些读者可能的确会定期处理此类数据。因为本书介绍的方法可应用于宽泛的课题，所以为了满足读者的不同需求，我们力求选择不同类型的数据来举例说明图的使用。下面来看 `USArrests` 数据集，它给出了 1973 年美国各州每 100 000 人中犯重罪者的逮捕率：

```
> attach(USArrests)
> head(USArrests) # 显示前6行,使用USArrests显示所有

           Murder Assault UrbanPop Rape
Alabama      13.2     236       58 21.2
Alaska       10.0     263       48 44.5
Arizona       8.1     294       80 31.0
Arkansas      8.8     190       50 19.5
California    9.0     276       91 40.6
Colorado      7.9     204       78 38.7
```

该数据集包括四个已命名变量及它们的值。有一列顶部没有变量名。最左边一列的值是行名（`row.names`），在本例中此列为州的名称。很多时候，行的名称仅仅是一个数字。

我们来探索该数据集吧。首先，关于谋杀逮捕率，看看带状图可以告诉你什么。尝试并思考一下，关于谋杀逮捕率，从带状图上可以了解到什么。逮捕率几乎相同还是完全不同？

它们是聚集在一起还是分散的？你的预期是什么？你可能已经得出了一些有趣的见解，但还是通过如下命令考虑点图进一步的功能：

```
> dotchart(Murder)
```

图 4-1 类似于带状图，图中显示了每个州的位置 (沿着 *x* 轴)。然而不同的是，每个州都有各自的“行”或者水平线。因此，没有重叠，也没必要抖动。

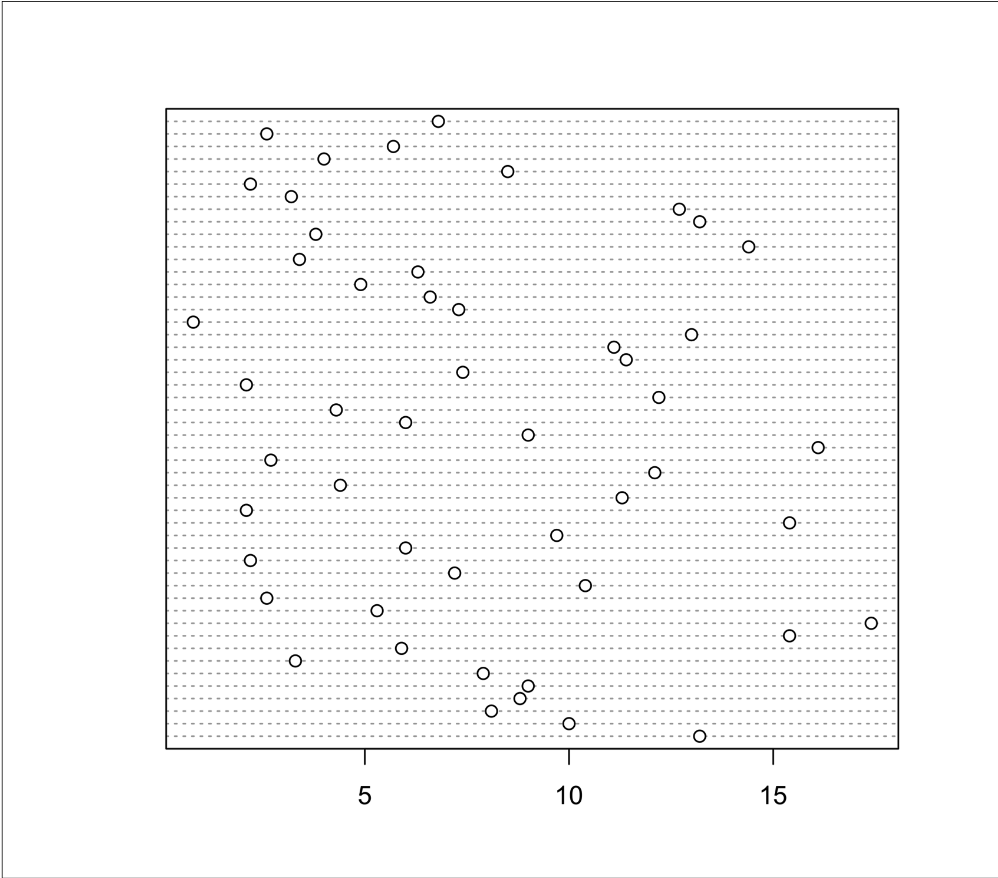


图 4-1：各州谋杀逮捕率的点图

可以对图形进行改进，例如为所有数据框加入字符向量，用 `row.names` 作为行标识符。请注意，数据框中的每一行有一个州名。通过添加参数 `labels = row.names(USArrests)`，可以用州名为点图的每一行添加标签。如果需要，标签也可以取其他变量的值，代码如下：

```
# 图4-2
dotchart(Murder, labels = row.names(USArrests), cex = .5)
```

由图 4-2 可以很容易地确定哪些州谋杀逮捕率最高，哪些州谋杀逮捕率最低，并找到一些典型的或几乎平均的逮捕率。参数 `labels` 把州名添加到图上；参数 `cex` 改变字符的大小。`cex` 的默认值是 1，它的值越小，字符就越小。

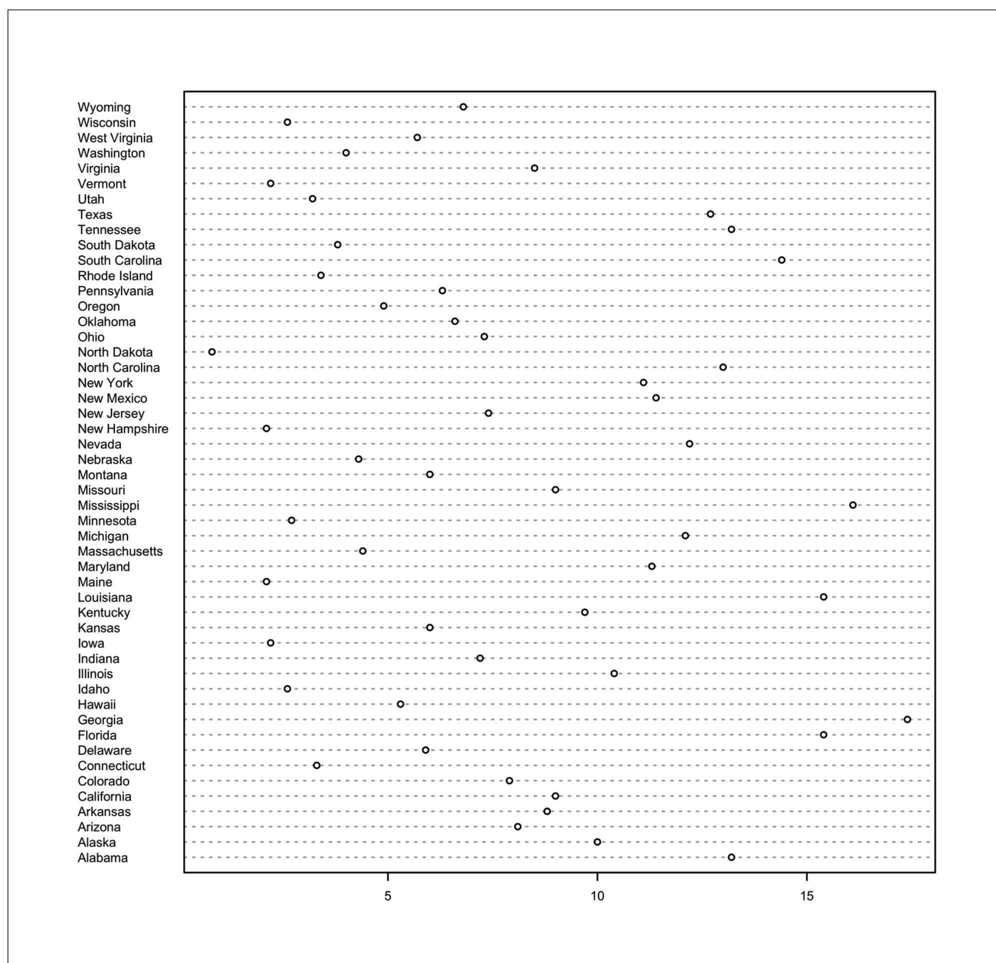


图 4-2：通过州来区分谋杀率的点图

查看该数据的一个更有趣的角度是查看按大小排列的谋杀逮捕率。要做到这一点，首先必须将数据按 `Murder` 的大小排序。这意味着数据集中的行将按照谋杀逮捕率重新排列。通过使用 `order()` 函数重新排序，可以创建一个新的数据集。可以用任意名称命名排序好的数据集，本例暂且称作 `data2`(此处没有创意奖)：

```
> data2 = USArrests[order(USArrests$Murder),]
```

接下来，使用新排序的数据重新绘图（见图 4-3），并添加标题和标签，代码如下：

```
> dotchart(data2$Murder, labels = row.names(data2),
    cex = .5, main = "Murder arrests by state, 1973",
    xlab = "Murder arrests per 100,000 population")
```

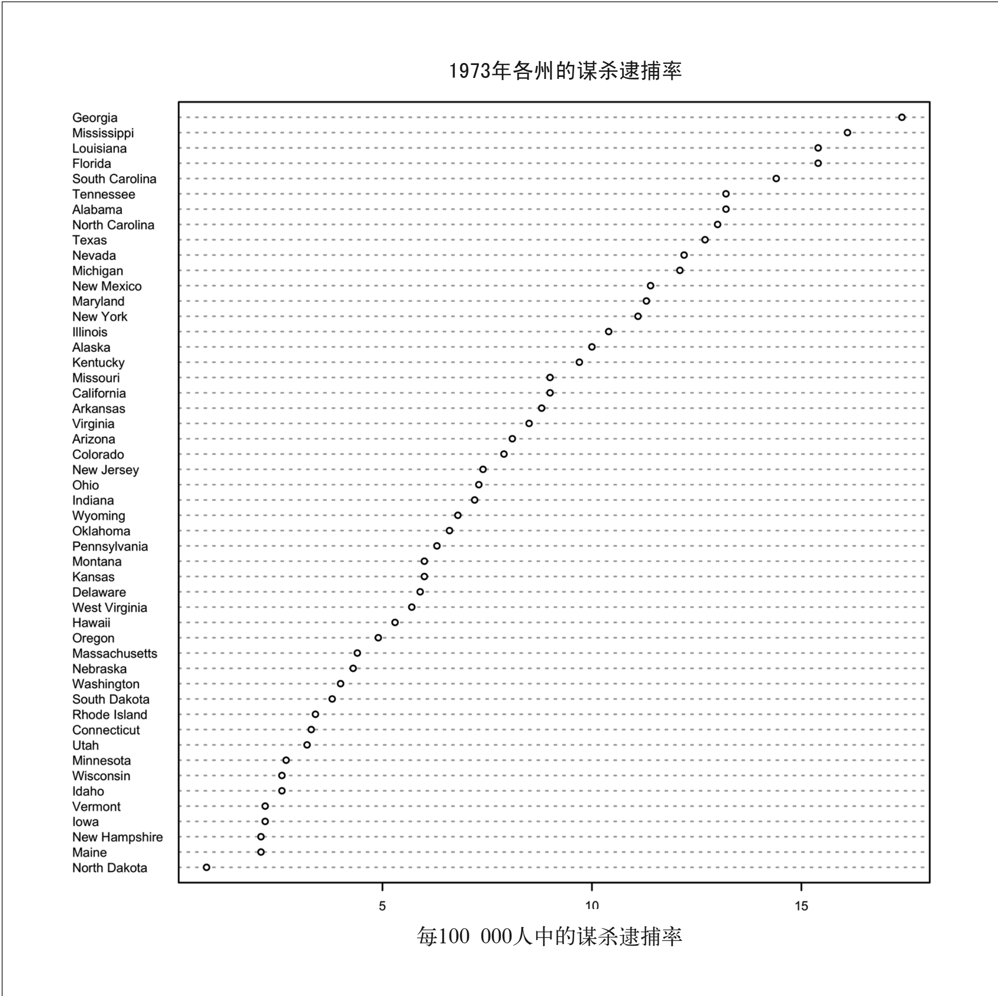


图 4-3：按照州谋杀率排序的点图

现在，很容易看到哪个州谋杀逮捕率最高，哪个州谋杀逮捕率最低。当然，可以从数据表中看出这些信息，但通过这张图，一眼便可看出各州之间的相对差异。这是你想象中的结果吗？记住，我们数据中的比率是逮捕率，而不是谋杀率。

通过一些细节调整，图形可能会更有吸引力。图符如果是实体的，将更加突出，因此添加 `pch = 19`。颜色可以吸引观察者的注意，所以用参数 `col` 将点和标签设置为不同的颜色。图中的线也靠得非常近，为方便阅读，尝试逐条线交替使用不同的颜色显示。使用参数 `col = c("darkblue","dodgerblue")` 可做到这一点。用 `lcolor = "gray90"` 将水平参考线设置成不同的颜色。

循环参数

许多 R 函数**循环**（recycle）使用参数。这意味着，如果向量中没有足够的项，R 将重复使用项。因此，绘制图 4-4 时，参数 `col = c("darkblue","dodgerblue")` 应用于 50 个州。因为只指定了两种颜色，所以当 R 需要为第三个州添加颜色时，它会返回使用第一种颜色，以此类推，直到所有州都有颜色。参数 `col` 可以包含任意数量的颜色。如果有 50 种或更多的颜色，R 将使用前 50 种颜色。如果颜色种类是少于 50 的任意数，那么 R 会重复，直到每个州有一种颜色。

使用下面的命令可以查看有哪些可用的颜色：

```
> demo(colors)
```

获得颜色名称列表的命令如下：

```
> colors()
```

附录 B 包含一个颜色表，以及创建图表的 R 代码。如果需要，可以打印一份。互联网上有两个不错的 R 颜色图表，可以访问 http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf 和 http://research.stowers-institute.org/efg/R/Color/Chart/ 查看。

图的标题越大越突出，所以添加 `cex.main = 2`，即把主标题放大一倍。完整的命令如下：

```
> dotchart(data2$Murder, labels = row.names(data2),
    cex = .6, main = "Murder arrests by state, 1973",
    xlab = "Murder arrests per 100,000 population",
    pch = 19, col = c("darkblue","dodgerblue"),
    lcolor = "gray90",
    cex.main = 2, cex.lab = 1.5)
```

结果如图 4-4 所示。

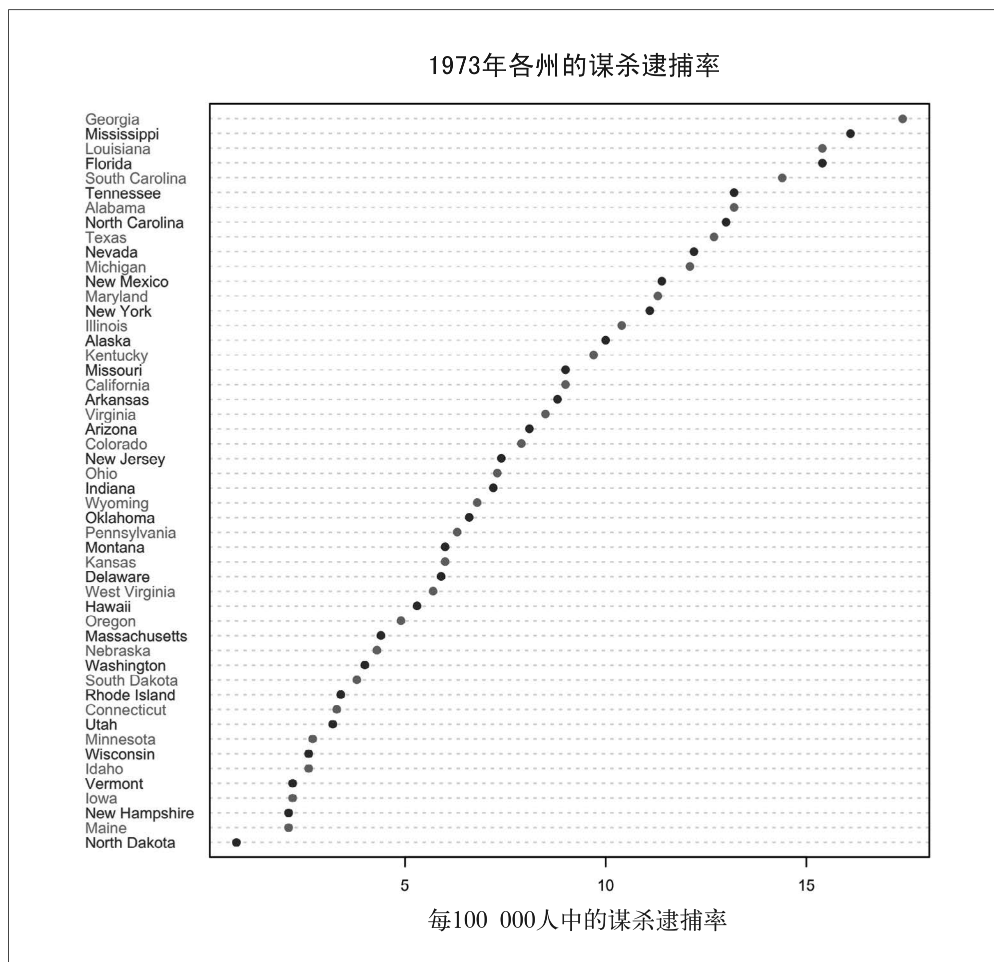


图 4-4：添加了颜色的州谋杀逮捕率的点图

练习4-1

为了理解为什么给图 4-2 中的图添加参数 `cex`，尝试不带该参数的 `dotchart()` 命令，看看会发生什么。

练习4-2

用 `Nimrod` 数据集中的变量 `time` 绘制点图。请记住，要先使用 `load()` 命令来加载数据。

第 5 章

箱线图

5.1 箱线图

有时，看一组数字的摘要信息或许比看数字本身更有帮助。有一种图通过按照定义好的数字范围将数据分组实现了这一点，这就是箱线图（box plot）。我们将用一个相对较大的数据集试着绘制箱线图。前面几种类型的图对于该数据集并不是非常合适。

`nlme` 包中有一些有趣的数据集，获取并加载该包的代码如下：

```
> install.packages("nlme")
> library(nlme)
```

下面看看 `MathAchieve` 数据集。超过 7000 行数据，这远远大于我们之前处理的数据集。如果想研讨 `MathAch` 得分的分布，这么大的数据量会给我们造成什么问题？让我们看看用带状图呈现该数据的情况。

在生成图 5-1 及后面许多例子的代码中，`par()` 中使用了参数 `mfrow`，作用是在一页内显示多个图形。格式为 `mfrow = c(i,j)`，其中，*i* 是图的行数，*j* 是图的列数。代码如下：

```
# 图5-1
library(nlme)
par(mfrow=c(2,1)) # 设置一图位于另一图之上:两行/一列

stripchart(MathAchieve$MathAch, method = "jitter",
  main = "a. Math Ach Scores,
  pch = '19'", xlab = "Scores", pch = "19")
```

```
stripchart(MathAchieve$MathAch, method = "jitter",
  main = "b. Math Ach Scores,
  pch = '.'", xlab = "Scores", pch = ".")
```

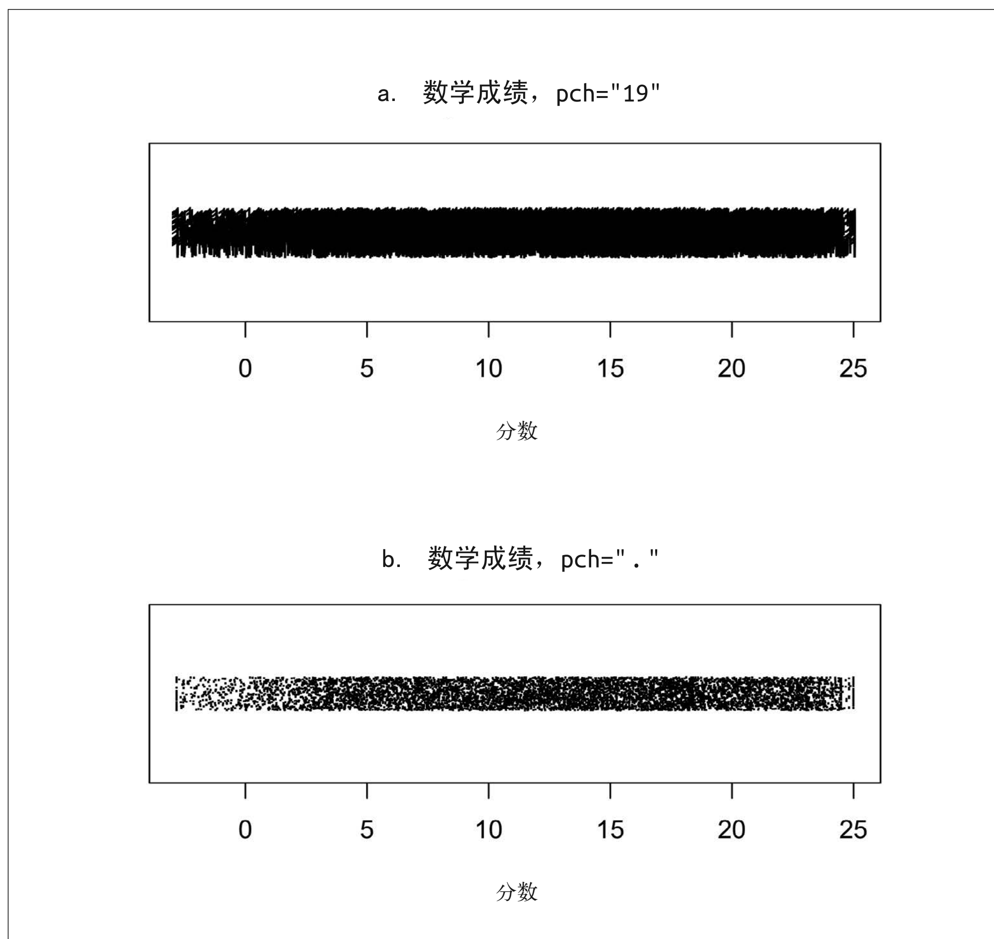


图 5-1：数学成绩的带状图

这两个带状图展示了使用图 3-4 中图形符号的结果（如图 5-1a 所示），以及使用另一种较小符号的结果（如图 5-1b 所示）。即便如此，图形还是很密集。有太多的点，所以很难判断分布的形状。中心在哪里？或许在约为 10 的地方，或者稍微高一点？分布有点偏斜（skewed），还是左边密度较低？有多少个极值点？不幸的是，帮助文件没有给出参考。对数学考试成绩做一些了解是很有必要的，因为有很多分数低于零分。

也许一种不同类型的图会揭示更多信息。箱线图的图形显示了几个关键的测量，它们在带状图中是不明显的。首先，让我们用下面的代码绘制一些箱线图。注意，这次参数 `mfrow` 在页面上创建了一个不同的一行两列的布局图，代码如下：

```
# 图5-2
library(nlme)
par(mfrow = c(1,2)) # 两图并排:一行,两列

boxplot(MathAchieve$MathAch,
  main = "Math Achievement Scores", ylab = "Scores")

boxplot(MathAchieve$SES,
  main = "Socioeconomic Status", ylab = "SES score")
```

这种图也称为盒须图（box-and-whiskers plot），如图 5-2 所示，中间的黑线代表中位数（median），该点将分数一分为二，一半较低，一半较高。由图可知，中位数约为 13。描述中位数还有其他方法，例如第 50 个百分位（50th percentile），该点处 50% 的分数较低；或是第二个四分位（second quartile），该点处二分之一的分数较低。盒子下面的边为第一个四分位（first quartile），该点处四分之一（即 25%）的分数较低。盒子上面的边是第三个四分位（third quartile），该点处四分之三（即 75%）的分数较低。从盒子垂直出来的线为"胡须"，如果点不超过四分位差（interquartile range，即第一个和第三个四分位数之间的差距）的 1.5 倍，须可达到最高点和最低点。如果点超出须的话，那么点呈现为小圆圈。

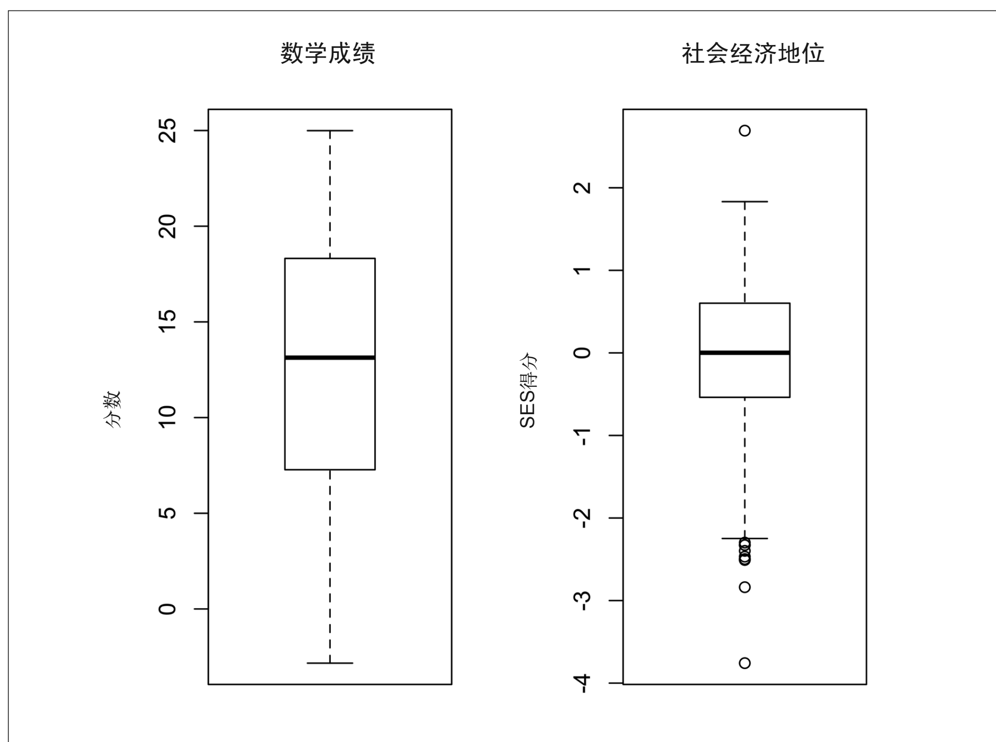


图 5-2：数学成绩和 SES 得分的箱线图

图 5-2 显示了数学成绩的分布几乎是对称的，但略偏向于较低分数；也就是说，下须稍长一点。

SES（社会经济地位）是数据集中的另一个变量，研究它是很有必要的。比较其箱线图（见图 5-2 右图）和数学成绩的箱线图，在 SES 图中有几个点超出了胡须。仔细研究，极端值总是会预示一些关于所研究变量和数据准确性的问题。你可能想检查极端值，以确保数据已输入正确。如果是这样的话，也许更应该查阅关于 SES 测量的文献中有关极值出现的原因，并思考为研究选取的样本的性质。

让我们回到数学成绩。把样本分成更小的组来比较考试成绩，这样可能会有启发作用。`Mathach` 被拆分为多个组，生成的箱线图如图 5-3 所示。`par()` 函数通过给参数 `mfrow` 传递一个表示两行和两列的向量，来设置一页显示四张图。通过参数 `sub = 'text to appear'`，使每张图的 x 轴上有一个标签，表明生成此图的命令。

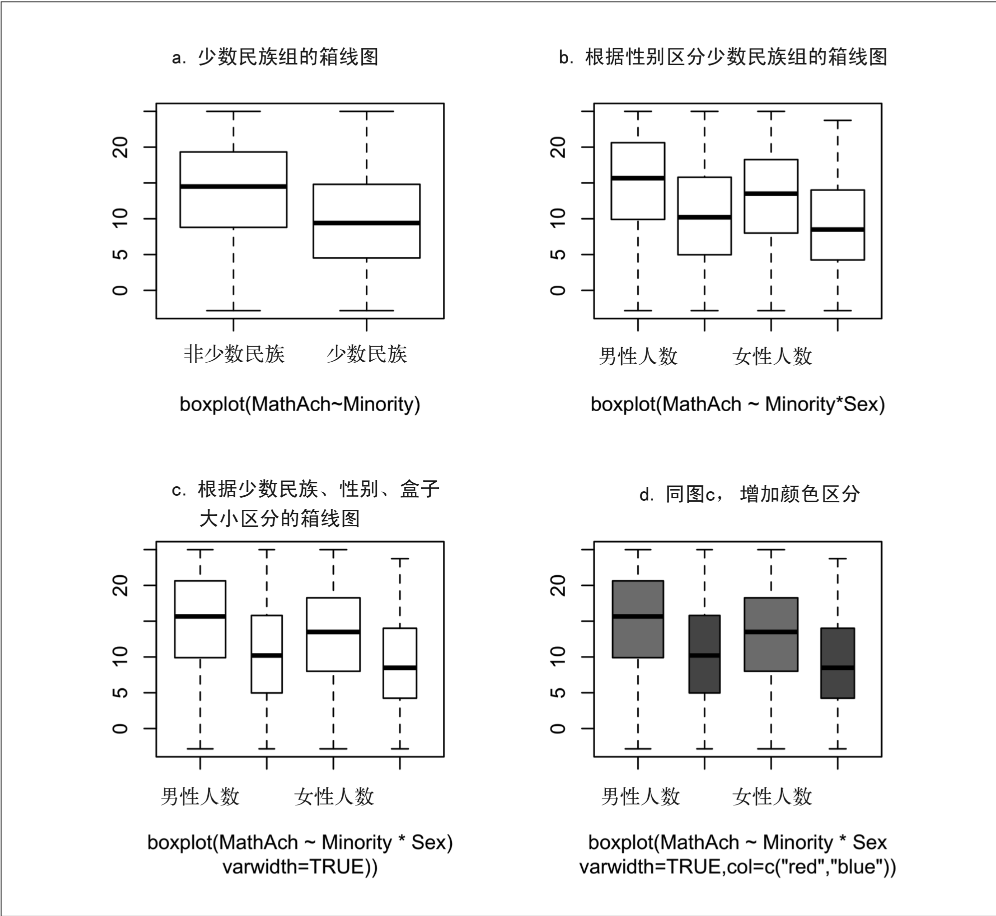


图 5-3：逐步细化的图之间的比较

生成图 5-3 的代码如下：

```
# 图5-3,即4张图
par(mfrow=c(2,2)) # 页面设置为2行,每行2张图
attach(MathAchieve)

boxplot(MathAch ~ Minority, xlab = "boxplot(MathAch~Minority)",
  main = "a. Math by Minority", cex = .4)

boxplot(MathAch~Minority*Sex,
  xlab = "boxplot(MathAch ~ Minority*Sex)",
  main = "b. Math by Min & Sex", cex = .4)

boxplot(MathAch ~ Minority * Sex,
  xlab = "boxplot(MathAch ~ Minority * Sex)",
  sub = 'varwidth=TRUE))', varwidth = TRUE,
  main = "c. By Min & Sex, width~size", cex = .4)

boxplot(MathAch ~ Minority*Sex,
  xlab = 'boxplot(MathAch ~ Minority * Sex',
  varwidth = TRUE, col = c("red","blue"),
  main = "d. Same as c. plus color",
  cex = .4, sub = 'varwidth = TRUE,
  col = c("red","blue"))')
```

下面来研究这四张图。图 5-3a 中，一个是非少数民族学生的箱线图，另一个是少数民族学生的箱线图。由图可知，虽然非少数民族学生得分的中位数和四分位数较高，但两组的最高分和最低分相同。图 5-3b 中，少数民族组又根据性别进一步分组。我们看到，不仅少数民族所有性别的数学成绩均较低，而且非少数民族和少数民族女性的成绩均较低。图 5-3c 和图 5-3d 都是图 5-3b 的改进版。在这些图中，每个盒子的宽度与它所代表的分组的大小有关。可以看出，较少的学生是少数民族，且在少数民族和非少数民族分组中，各组的男性均比女性少。图 5-3d 运用色彩使分组更容易区分。颜色向量仅指定两种颜色，这两种颜色赋给前两个盒子图，然后将相同的两种颜色*循环*赋给后面的两个盒子图。这样循环的效果是非少数民族男性和女性为浅灰色，少数民族的男性和女性为深灰色。（如我们在前一章所看到的，R 循环使用任何不够长的向量以便完成给定的任务。）

5.2 再次访问Nimrod

现在是重回 `Nimrod` 数据集的好时候。我们可以用箱线图来研究用各种乐器及业余与专业剧团的表演时间的分布情况。此外，可以看到一个前面未提及的重要信息。让我们从 `time` 的箱线图开始，按 `level` 和 `medium` 分组，代码如下：

```
# 图5-4a,首先必须加载Nimrod
# load("Nimrod.rda") / 见第1章
attach(Nimrod)
par(mfrow=c(2,1)) # 图片在同一页面中展示 / 两行一列
boxplot(time ~ level * medium,
  main = "a. Performance time by level and medium")
```

结果如图 5-4a 所示，各组有相当大的变化。除了铜管乐队以外，专业团体往往比业余团体演奏的速度更慢。在该图的下一个版本中，用颜色突出业余与专业的状态差异，可能会更清楚。由于专业团体演奏的速度一贯较慢，这种差异可以表明“正确”的节奏是比较慢的吗？这一点明显值得思考。

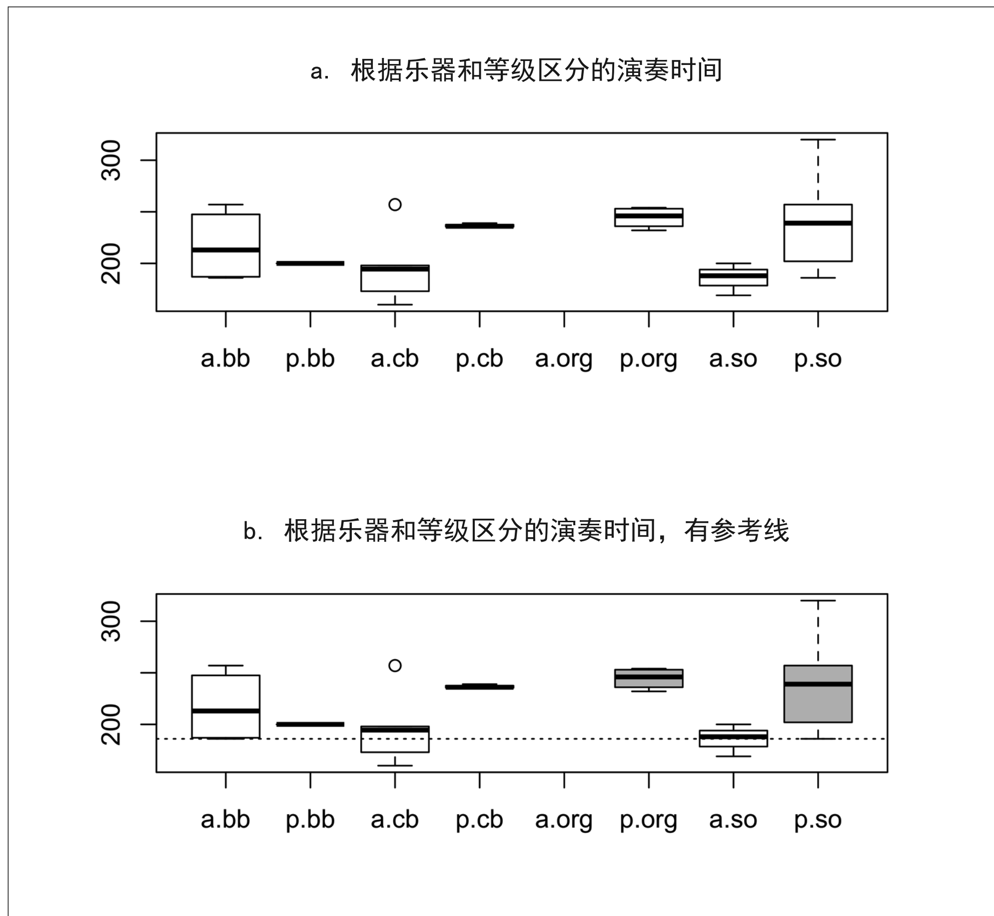


图 5-4：根据乐器和等级绘制的 Nimrod 演奏时间图

邮
电

现在是时候揭开数据集中秘密编码的表演者的身份了。

“EE”正是爱德华·埃尔加（Edward Elgar），是作曲家本人！尽管在样本中有训练有素的专业音乐家的精湛表演，我们仍可以合理地主张，关于该作品的合适的节奏，作曲家的演奏应该更权威。当然，我们只有作曲家的一次表演的数据，并不知道他每一次演奏是否都是这个节奏，或接近这个节奏。尽管如此，与图上的其他所有组对比，也可能会揭示爱德华的节奏。为此，可以添加一个参考线（reference line）。参考一开始的电子表格（见表 1-4），发现爱德华的表演时长为 186。在 186 的地方加一条横穿所有箱子的水平线，以使

比较每个作曲家的表演时间变得更加容易。可以用 abline() 命令实现这个操作，它将立即在如下命令之前生成的图上添加一条线：

```
# 图5-4b
boxplot(time ~ level * medium,
  main = "b. Performance time by level and medium with reference
  line", col = c("white", "light blue"))
abline(h = 186, lty = "dotted")
detach(Nimrod)
```

在图上添加线

可以使用 abline() 函数来给图添加一条或多条**参考线**（可以根据需要多次使用该函数，在一张图上添加多条线）。此函数只能绘制直线，绘制曲线采用的是不同的函数。线可以通过它的截距和斜率定义（如果你不记得高中数学中截距和斜率的含义，在第 12 章有简要回顾），或者像在 Nimrod 的例子中那样，只给出 y 截距来画一条水平线（h = 186，即 time = 186）。在适当的地方，可能用 v = 而不是 h 来请求绘制垂直线。可以调用各种类型的线，比如本例中使用 lty="dotted"。获取更多信息，请输入 **?abline** 或者 **?par** 查询。

图 5-4b 表明，业余音乐家比专业音乐家更认同作曲家的节奏。这是一个令人惊讶的结果，会生成一张令人兴奋的图。当然，我们只有一个小的表演样本。如果我们的目的是概括专业的和业余的表演，那么目前的样本是不够的。然而，我们只是在探索可能的关系，并假设可能希望进一步调查研究，这种情况下数据就足够了。

5.3 美化数据

图 5-4b 完成了展示 time 与 level 及 medium 关系的基本工作。如果这就是你想要的，可以直接跳到本章结尾的练习。之后，当你感觉可以更自如地编写 R 代码，并想要一个更具吸引力的图形时，可以回到本节。

我们可以让 Nimrod 的图更有吸引力，也更容易理解。图 5-4b 最令人不满意的一点是乐器组的名称，例如“a.bb”。像“业余铜管乐队”（amateur brass band）这样较长的名字，在有限的空间里根本就不适合。解决这个问题的方法之一是水平放置条（bar），在空白处为每个组名空出一行。同时为乐器分组提供有意义的名称，并在图上放一些文本。新图呈现在图 5-5 中。

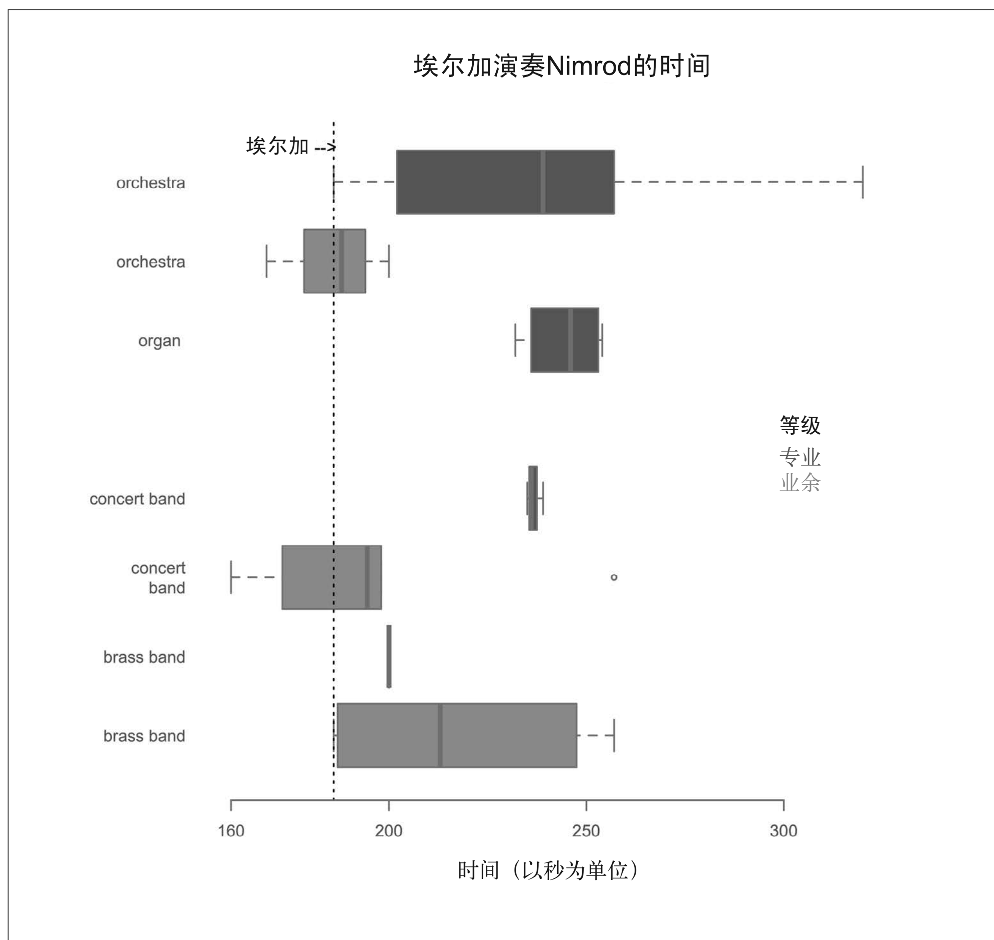


图 5-5：根据 `medium` 和 `level` 改进的 Nimrod 演奏时间图，将其与图 5-4 对比

你同意图 5-5 是图 5-4b 的改进版吗？如果同意，我们来看一下怎么改进。

生成新图的脚本如下。有几个命令被空白行分隔开，以使其更容易阅读。每个命令上方的注释，解释了参数的含义：

```
# 图5-5的脚本
attach(Nimrod)

# par() 设置背景色、前景色、坐标轴颜色、
# 文本大小(cex)、水平展示、
# y轴(las=1)的文本,以及页边空白(mar)。对于默认边距来说,图像太大。
# 关于以上参数,输入?par可获得更多信息。

par(bg = "white", fg = "white",
  col.axis = "gray47", mar = c(7,8,5,4),
  cex = .65, las = 1)
```

```
# boxplot() 确定公式 (time ~ level * medium),
# 使图水平扩展,
# 为盒边界和盒(col)设置颜色,
# 创建标题(main,xlab),
# 为组合level*medium创建名字(names)和名字大小(cex.names)。
# 一个名字为"",
# 因为没有"amateur organ"这一类别。

boxplot(time ~ level * medium, horizontal = TRUE,
  border = "cadetblue",
  main="Performance Times of Elgar's Nimrod",
  col = c("deepskyblue","peachpuff4"),
  xlab = "Time in seconds",
  names = c("brass band","brass band","concert
  band", "concert band","", "organ ", "orchestra","orchestra"),
  cex.names = .4)

# abline()在时间为186秒处放垂直线来展示
# 由爱德华指挥的演奏，线的类型(lty)为点线,
# 且颜色(col)为黑色。

abline(v = 186, lty = "dotted", col = "black")

# legend() 选择图例的文本、颜色和在图片上的位置。
# 图例中,专业组显示为桃色,业余组显示为深天蓝色。

legend("right", title = "Level", title.col = "black",
  c("Professional","Amateur"),
  text.col = c("peachpuff4","deepskyblue"),
  text.font = 2, cex = 1.2)
# mtext() 将文本放在用户指定的位置
mtext(" Elgar himself - - >", side = 3,
  line = -2, adj = 0,
  cex = .7, col = "black")

# axis()修改了x轴(1),设置了颜色、长度和刻度线
axis(1, col = "cadetblue", at = c(160,200,250,300))

detach(Nimrod)
```

符号是什么意思？ legend() 函数

图例（legend）是图上的一个标记，表明特定符号或颜色的含义。例如在图 5-5 中，右边的图例规定，褐色（R 的 peachpuff4，书中显示为深灰色）盒子代表专业人士的合奏时间，而蓝色盒子（书中为浅灰色）代表业余团体的演奏时间。legend() 函数允许在图上添加位置说明、特定文字、颜色、字体等。欲知更多信息，请输入 **?legend**。

通过图 5-4 和图 5-5 可知，使用 R 可以生成基本的图表，来简单快速地探索数据（通常用一行代码）。如果需要很漂亮且用于演示的图，R 也能实现（但可能要做更多工作）。

5.3.1 练习5-1

正如前文所述，关联不能证明因果关系。图 5-3 中，数学成绩与少数民族状态间明显的关系，实际上可能是其他因素的函数。用 `Minority` 和 `Sex` 分组，对变量 `SES` 进行箱线图分析。保存你的图（关于如何保存图，见第 2 章），以便在以后的分析中使用。使用字处理程序，用一段或两段文字说明你的发现，并插入图说明你的观点。在稍后的练习中，我们将直接研究数学成绩和 `SES` 之间的关系。

5.3.2 练习5-2

比较两个或两个以上分组时，一种可替代箱线图的图形是 Engelmann-Hecker-Plot（EH 图）。比较两个图：一个是将 `cyl` 作为分组变量的、为 `mtcars` 数据集中的变量 `mpg` 制作的箱线图（参见练习 3-1），另一个是用 `plotrix` 包中的 `ehplot()` 函数创建的 EH 图。EH 图有哪些优点？箱线图有哪些优点？

第 6 章

茎叶图

基础茎叶图

一些人可能会认为这简短的一章是一种“怀旧”，因为它描述了一种在用纸和笔分析数据的时代非常重要的图。在现代的演示中，你可能不会看到很多这类图的例子。之所以把它包含在本书里，是因为它有助于你更好地理解直方图（第 7 章的主题）。也许你会发现它在数据检验的探索阶段也是有帮助的。如果你已经了解了直方图，可以跳过本章，不必担心错过必要的资料。

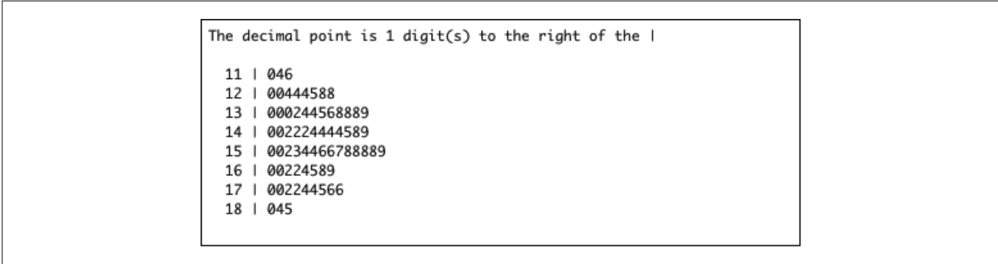
`multcomp` 包中的 `sbp` 数据集包括变量 `sbp`（69 名患者的收缩压）以及每个人的性别和年龄。我们可以用茎叶图（stem-and-leaf plot）观察血压的分布。这种类型的图不仅可以揭示数据分布的大致形状，还可以显示每个数据点的（四舍五入）值。

茎叶图按照从低到高的顺序排列所有的值。然后，为同一范围内的所有值保留一行，并且在适当的行上写下每个数值的最后一位有效数字。可以使用基础 R 中的 `stem()` 命令为 `sbp` 数据集里的变量 `sbp` 创建一个茎叶图。这种类型的展示，有时也被称为文本显示（textual display），它出现在 R 控制台而不是图形窗口中。生成图 6-1 的代码如下：

```
> library(multcomp)
> stem(sbp$sbp)
```

图 6-1 显示了数据集中所有的血压。左边的列（包括数字 11、12、13 等）包含的是“茎”。血压都是三位数，所以“茎”包含前两位数字，“叶”包含每个血压值的最后一位数字。来看第一行，第一个“茎”代表的数字以 11 开头且树叶上的数字是 0、4 和 6。因此，第

一行代表的数据有 110、114 和 116。下一行“茎”包括数字 120、120、124、124、124、125、128 和 128。我们可以看到，收缩压有两个 170 和四个 158，但只有一个 185。


```
The decimal point is 1 digit(s) to the right of the |

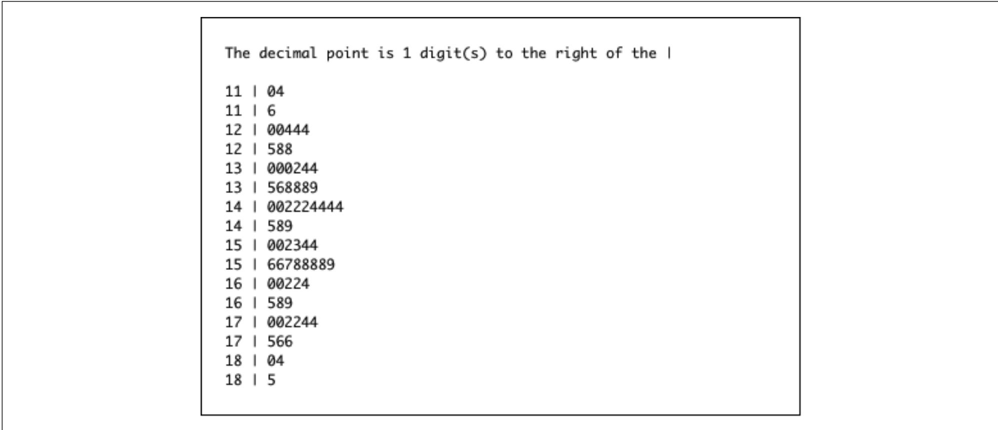
  11 | 046
  12 | 00444588
  13 | 000244568889
  14 | 002224444589
  15 | 00234466788889
  16 | 00224589
  17 | 002244566
  18 | 045
```

图 6-1：变量 sbp 的茎叶图

由图 6-1 可知，数据的分布是近似对称的。低血压和高血压的数量大致相同，而且大量的血压分布在中心附近（即 130~160 的血压）。在这个图中，“茎”的宽度大约是 10（例如 110~119、120~129）。如果“茎”的宽度改变了，图的形状很可能也会改变。可以为 `stem()` 命令添加另一个参数，来改变茎杆的宽度。参数 `scale = ` *x* 控制每个“茎”的宽度，其中 *x* 是一个正数。例如，试试如下命令：

```
> stem(sbp$sbp, scale = 2)
```

结果如图 6-2 所示，该图的长度是之前的两倍。每个茎的宽度减半（即宽度为 5 而不是 10），而茎的数量变成之前的两倍。大体的分布形状没有改变，但中心略有下降，大约在 145 处，之前我们并没发现这一点。在一些分布中，宽度的改变会显著改变形状。在直方图中，叶的宽度被称为区间（bin）尺寸。也许茎叶图看起来没有其他一些有好看形状和颜色的图吸引人，但此类图显示了所研究向量中的每个数字的精确值，这可以帮助理解数据，也有助于对图的修改。


```
The decimal point is 1 digit(s) to the right of the |

11 | 04
11 | 6
12 | 00444
12 | 588
13 | 000244
13 | 568889
14 | 002224444
14 | 589
15 | 002344
15 | 66788889
16 | 00224
16 | 589
17 | 002244
17 | 566
18 | 04
18 | 5
```

图 6-2：sbp 的茎叶图，茎长扩大一倍

茎叶图最适合中小规模的数据集。如果一个茎中有太多的数据，它们将会超出页面！当这种情况发生时，我们就失去了对真实的分布形状的感知。解决该问题的方法之一是加大 `scale`，让叶子变多，而每个叶子中的数据变少。有时这种策略也帮助不大。你也许不会将这种类型的图用在最后的展示中，但可能会发现这个优雅的工具有助于理解直方图，而且在项目的探索阶段它是有启发作用的。

练习6-1

亲自为血压数据选择一个合适的 `scale`，所选数字不需要是整数，也可以比 1 小。尝试用前面章节中研讨的数据集执行相同的分析。

练习6-2

在同一张图中对比两个变量的分布有时很有帮助。可以用 `aplpack` 包（需要安装并加载该包）中的 `stem.leaf()` 函数绘制两个背对背的茎叶图。用 `trees` 数据集中的变量 `Height` 和 `Volume` 执行该操作。可以看到预期的结果吗？为什么有或为什么没有？（提示：每个变量的测量单位是什么？）使用 `car` 包中 `Baumann` 数据集里的 `pretest.1` 和 `post.test.1` 尝试绘制相同的图。后一个测试比之前的测试是高还是低？这是一个有帮助的工具吗？

第 7 章

直方图

7.1 简单直方图

我们来重温一下 `multcomp` 包中的 `sbp` 数据集。对于分析血压数据，一个有用的工具就是我们熟知的直方图（histogram）。在这种类型的图中，通常把所要研究的数值型变量（如 `sbp`）的值的范围放在水平标尺（*x* 轴）上。这个范围被分成几部分，称为区间（bin）。垂直标尺（*y* 轴）显示每个区间中落入多少观测值。

图 7-1 是通过调用 4 次 `hist()` 函数生成的，每调用一次生成一张图。当然，你可以在控制台输入几行命令来完成这一操作。基本的命令仅有 `hist(sbp)`。

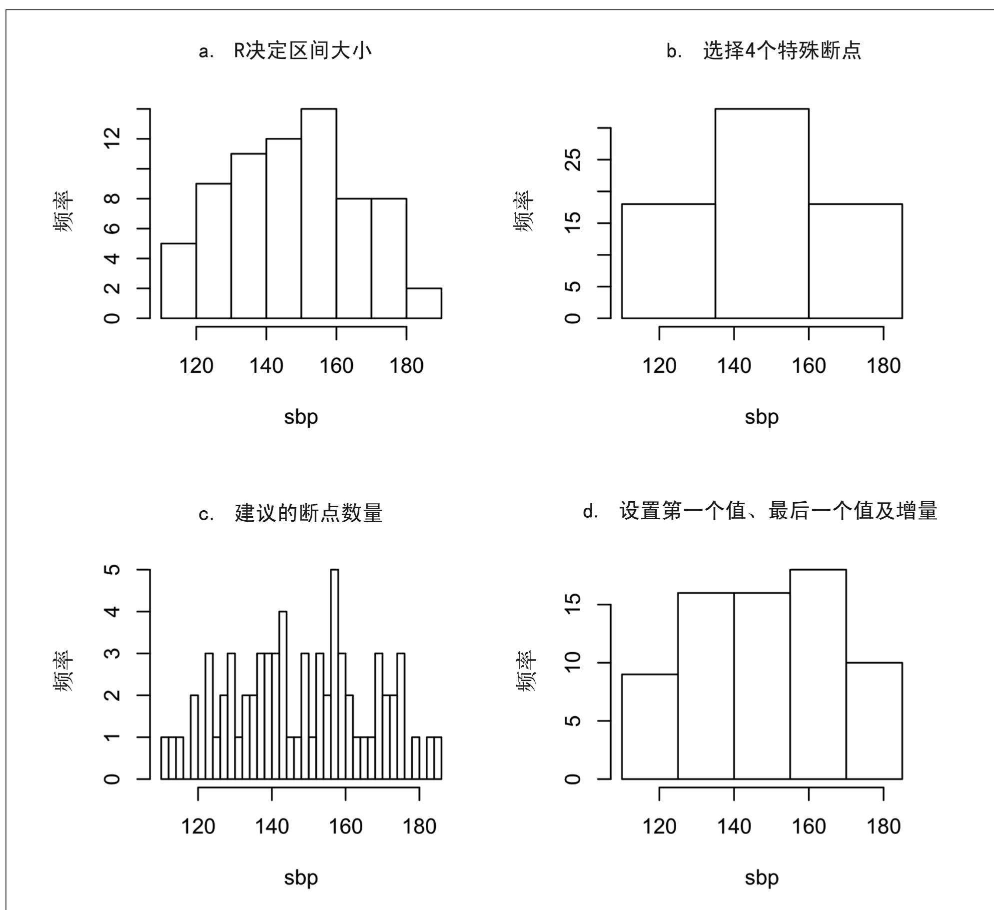


图 7-1：sbp 数据集中变量 sbp 的直方图，比较不同区间中数据的效果

生成图 7-1 的脚本如下：

```
# 图7-1的脚本
library(multcomp)
attach(sbp)
par(mfrow = c(2,2))
hist(sbp, main = "a. Let R decide bin sizes")
hist(sbp, main = "b. Choose 4 particular breakpoints",
  breaks = c(110,135,160,185))
hist(sbp, main = "c. Suggest number of breakpoints",
  breaks = 30)
hist(sbp, main = "d. Set first, last and increment",
  breaks = seq(110,190,15))
detach(sbp)
```

这段代码把区间数量的选择留给了 R，这通常是一个很好的决定。此外，每个命令行添加参数 `main =`，为特定直方图生成标题。最后，除了第一张图，参数 `breaks =` 表明使用区间

的数量和 / 或区间的宽度。注意，在此处创建的所有图形中，区间的大小都一样；换句话说，条的宽度都是一样的。这一点是直方图的惯例，不这样做的话就很难解释。

可以使用参数 breaks 执行如下操作。

- 在条之间提供许多断点，例如 breaks = 30。
- 定义详细的断点，例如 breaks = c(110,135,160,185)。
- 给出第一个值、最后一个值以及增量，例如 breaks = seq(110,190,15)。
- 提供一系列数字的任意其他有效规格，例如 breaks = c(110:190)。

使用 breaks 时，要从向量中的最小值开始；否则，会得到错误信息。在前面的例子中，sbp 中的最小值是 110。

图 7-1 是 sbp 数据的直方图的几个例子。它们都是关于相同变量的直方图，但看起来完全不同，这是因为图中的区间数量不同。关于 sbp 得分的分布，从不同的直方图中可能得出不同的结论。

图 7-1a 给人的印象是，许多患者的 sbp 分数约为 150~160，但少量患者有更高和更低的分数。此外，分布也不均匀。而图 7-1b 给人的印象是分布完全对称，此图是通过使用参数 breaks = c(110,135,160,185) 创建的。图 7-1c 是通过参数 breaks = 30 创建的。相比图 7-1b，图 7-1c 更像图 7-1a，但它比其他两张图展示了更多细节；它也更不稳定，有更多剧烈的起伏。图 7-1d 是利用参数 breaks = seq(110,190,15) 创建的，似乎和图 7-1a 相像，但在高端（近 180 处）没有显示血压的大幅下降。

图 7-1a 使用了默认的区间数，如果你不指定一个值，R 就选择默认的区间数。相比图 7-1b，图 7-1a 的分辨率更高，它有更多的条，能更清晰地理解数据下降的位置，但清晰程度比图 7-1c 差。综观所有的直方图，看起来缺乏对称性且在高端下落，这两个重要特征正是我们想让数据展示出来的特点。在这个例子中，默认的选项（见图 7-1a）给出了一个很好的结果，事实上通常但并不总是如此。练习 7-1 就是一个反例。

你可以控制直方图中区间的数目，但没有希望的那样容易。如果指定了 break 的数目，如图 7-1c 一样，这只是作为一个建议。R 将选择一个可能接近，但符合“漂亮”标准的数字。关于这个规则的更多信息，请输入 **?pretty** 查询。

可以给 hist() 命令添加许多其他参数。其中一些将体现在下面的例子中，如图 7-2 所示。例如，las = 1 可旋转 *y* 轴上的数字，使它们垂直于水平线，label = T 在每个条的顶端添加频率数，col="maroon" 设置了条的颜色，参数 xlab = 为 *x* 轴提供了更具描述性的标签。代码如下：

```
# 图7-2
hist(sbp, main = "sbp dataset", las = 1, label = T,
  col = "maroon", xlab = "Systolic blood pressure")
```

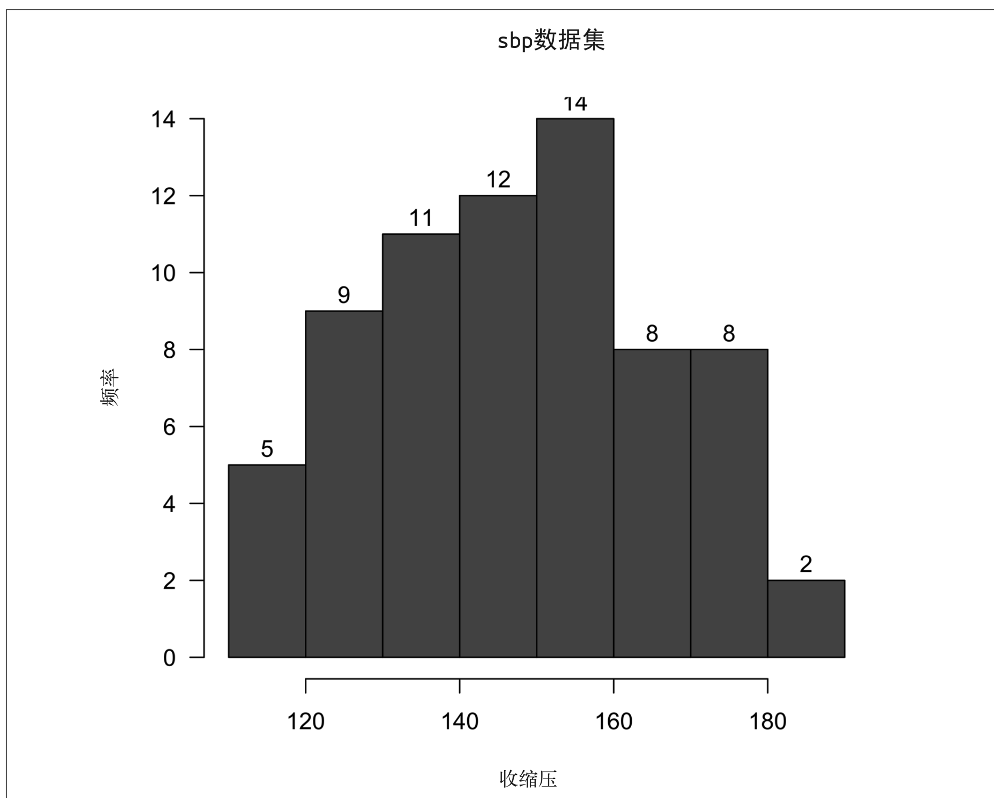


图 7-2：带有附加特征的直方图

7.2　带第二个变量的直方图

有时，进行比利用简单直方图更深入的数据研究十分重要。例如，了解不同性别的人的血压分布是相似还是不同。评估这种可能性的一种方法是使用堆叠直方图（stacked histogram）。sbp 数据集包含变量 gender，可以在堆叠直方图中使用性别区分图 7-1a 中的每个条，来显示有多少观测值是男性的，多少观测值是女性的。虽然 hist() 函数不支持这个操作，但 plotrix 包中提供的 histstack() 函数可以实现，代码如下：

```
# 图7-3
library(plotrix)
library(multcomp)
histStack(sbp$sbp, z = sbp$gender,
  col=c("navy","skyblue"),
  main = "Systolic blood pressure by gender",
  xlab = "Systolic blood pressure", legend.pos = "topright")
```

结果如图 7-3 所示。

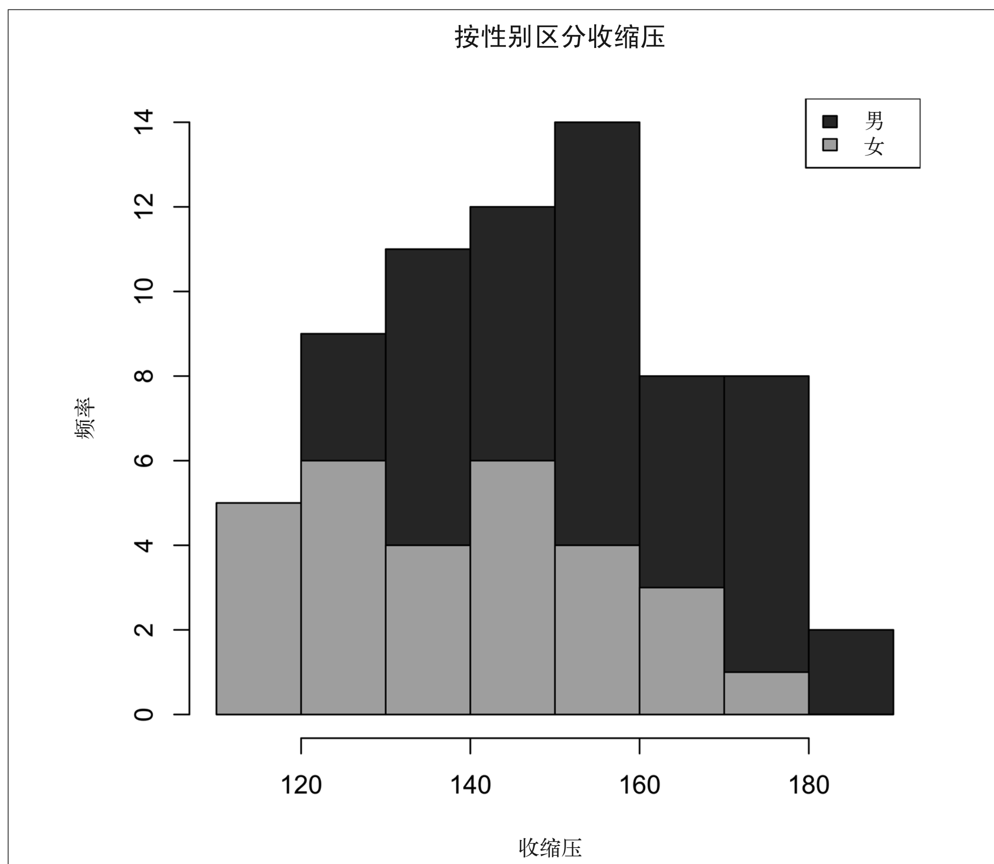


图 7-3：按性别区分的收缩压的堆叠直方图

堆叠直方图易于提供很多有用的信息。首先，用浅灰色表示的女性血压的分布一目了然。然而，男性的分布不那么容易解释：因为柱形条的底部在不同的水平线上，很难比较它们的高度。尽管如此，仍可看出男性比女性的血压更高，反之亦然。还有另一种方式来呈现此数据，大多数人觉得此种方式更容易阅读。也就是可以把信息分成并排或一上一下的两张图，而不是把所有的信息放在一张图中。在 R 中实现这个需求有多方法，最简单的方法之一是使用 RCmdrMisc 包中的 Hist() 函数，代码如下：

```
# 图7-4
library(RcmdrMisc)
library(multcomp)
Hist(sbp$sbp, groups = sbp$gender,
  main = "Systolic blood pressure by gender",
  col = "navy ", xlab = "Systolic blood pressure")
```

结果如图 7-4 所示。

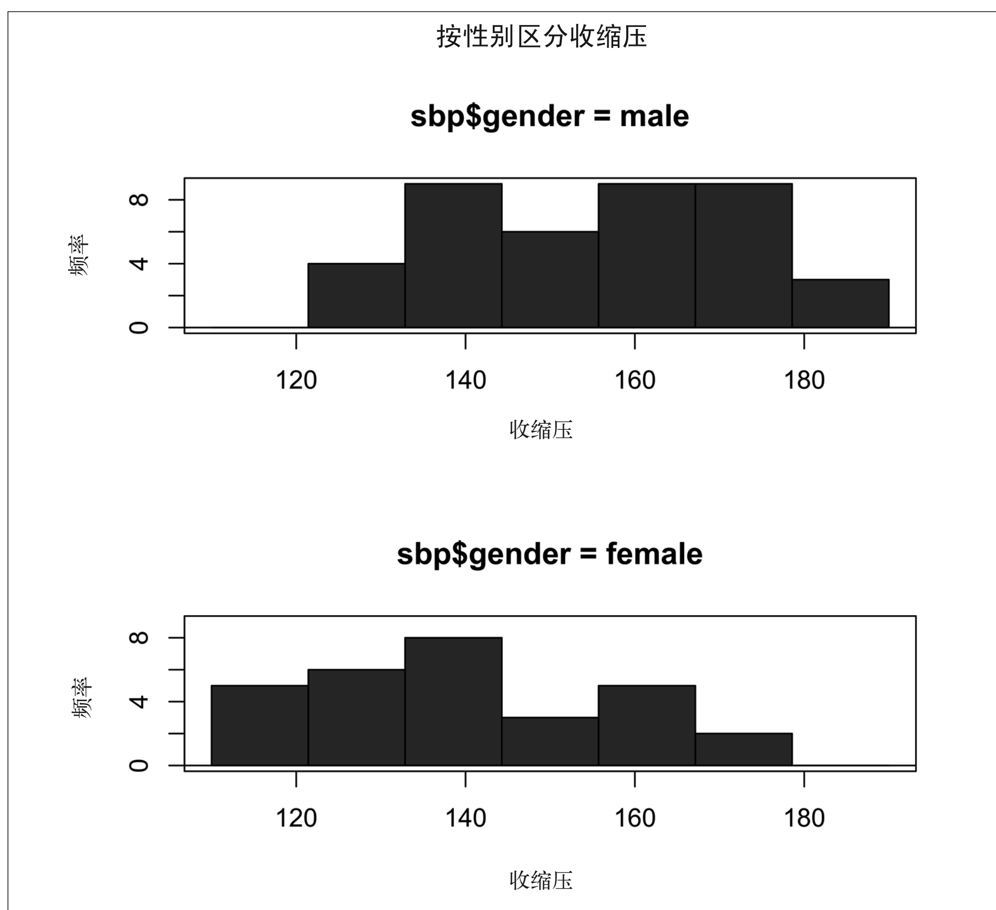


图 7-4：男性（male）和女性（female）收缩压分开展示的直方图

图 7-4 更清晰地显示出，男性不仅有最高的血压，而且集中在高端。对于数据量小的分组来说，`hist()` 函数很好用，但对于数据量大的分组却不那么方便。因此，我们可以转为使用 `lattice` 包。

`lattice` 包对于格子图（trellis graphic）有非常好的展现形式，图形根据观测值组进行划分，即为每个组呈现为单独的图形。`car` 包中的 `Salaries` 数据集包括 2008—2009 年美国一所大学的教师 9 个月的工资数据。收集数据的目的为了研究男女工资之间的差异。我们来生成一组直方图，为每个等级（3 个等级）和性别（2 种性别）的组合，或者说共 6 种组合，生成各自的直方图。首先，安装并加载必要的包。然后，查看数据集中提供的信息。注意 `histogram()` 命令的语法稍微不同，`Salaries$salary` 变量之前有波浪号（`~`）。由变量组合形成的组后紧跟竖线符号（`|`），使用星号（`*`）表示跨两个变量。图 7-5 非常易读，可以很容易地比较男性教师和女性教师的工资。分组变量的顺序是很重要的。如果颠倒了顺序，就会导致结果图不方便阅读。尝试以下代码：

```
# 图 7-5
install.packages("lattice") # 有可能已经安装了
install.packages("car")
library(lattice)
library(car)
head(Salaries)

       rank discipline yrs.since.phd yrs.service  sex salary
1      Prof          B            19          18 Male 139750
2      Prof          B            20          16 Male 173200
3      AsstProf      B             4           3 Male  79750
4      Prof          B            45          39 Male 115000
5      Prof          B            40          41 Male 141500
6      AssocProf     B             6           6 Male  97000

histogram(~ Salaries$salary | Salaries$rank * Salaries$sex,
  type = "count", main = "Faculty Salaries by Rank & Gender")
```

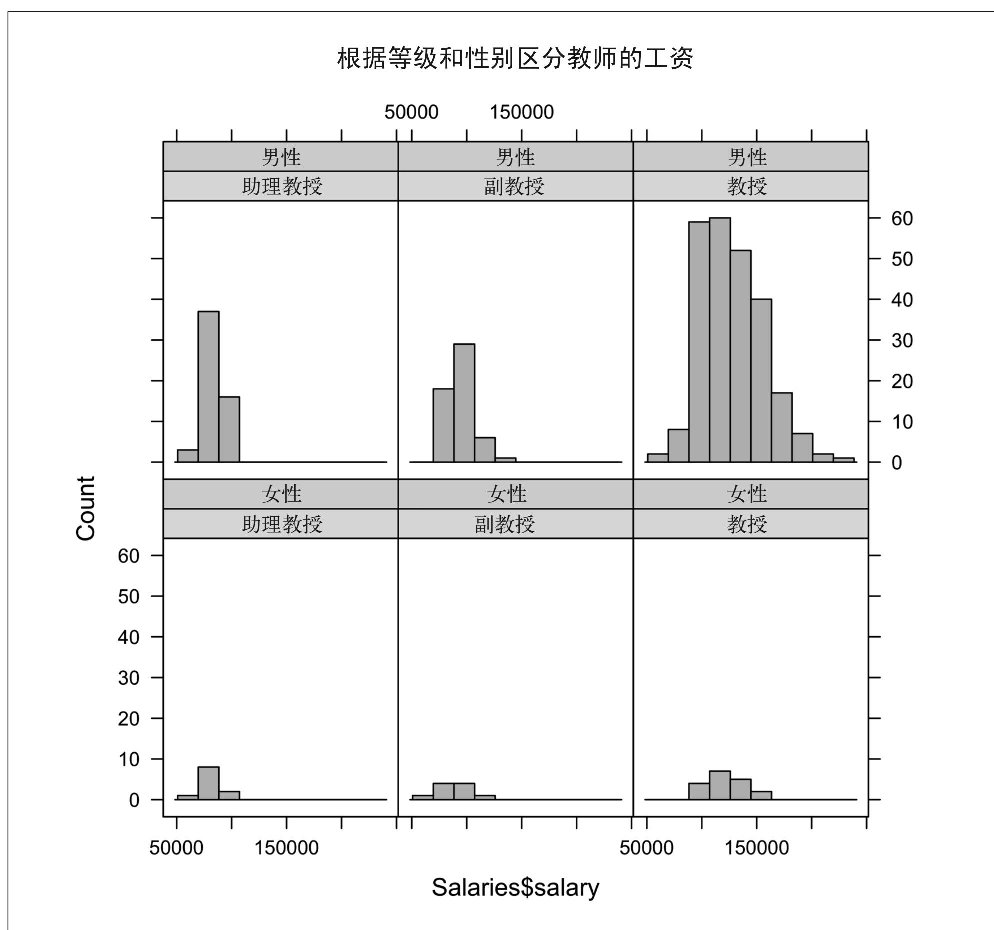


图 7-5：分组直方图，使用 lattice 包中的 histogram() 函数生成

从图 7-5 所呈现的数据中，可以观察出许多结果。首先，比较顶部显示的男性数据和底部的女性数据，可以看出，该项研究中男性比女性多出很多。其次，从左到右观看直方图，很显然，教授（最高等级）比副教授或助理教授更多。最后，每个等级中男性和女性的工资分布有相同的中位数，但在教授和副教授级别中有更多的男性处于工资的高端。

7.2.1 练习7-1

考虑来自 `Sleuth2` 包的 `case0302` 数据集。不指定 `breaks` 的个数，绘制变量 `Dioxin` 的直方图。接下来，尝试以几种不同的方式来创建各种尺寸的区间。分布的形状看起来有变化吗？再看看 `Dioxin` 的带状图，与其直方图比较，它会告诉你什么？

7.2.2 练习7-2

有时，你可能想比较两个变量。有很多方法可以实现，一种方式是观看它们的直方图。如果两个变量都由同一尺度测量，查看背对背的直方图也许会给我们启示。`Hmisc` 包中的 `histbackback()` 函数刚好能实现这个目的。你可以使用该函数研究 `car` 包中 `Burt` 数据集里兄弟智商的测量值间的关系。此外，也可以比较 `Salaries` 数据集中男性和女性的工资。

第 8 章

核密度图

8.1　密度估计

在科学领域中，一个常见的问题是从数据样本中估计出一个数学函数，描述一个变量（例如前面两章中 `sbp` 例子中的收缩压）取某个值的相对可能性。在第 7 章中，我们试图用直方图对这样的图做了粗略估计。如果看一下图 7-2 中的直方图，你可以看到很可能出现接近 150 的收缩压，但大约 110 的收缩压却不太可能出现。描述给定值（例如某个血压值）的可能性的规则或公式被称为*密度函数*（density function）。

对于很多问题来说，直方图是一个很好的工具，它容易理解且易于计算。然而，你应该意识到它也有缺点。许多函数是*连续的*（continuous），也就是说，函数可以取一定范围内的任意值。血压值可以是 120、123 或 129.2，但直方图可能会强制所有这些值在同一个区间（bin）内，从而使所有值都取 120。（在图 7-2 中，直方图区间的宽度为 10，那么所有大于等于 120 且小于 130 的数值会在同一区间内。）也就是说，我们用一个*离散*（discrete）函数——只能取所选的血压值——来估计连续的密度函数。图 8-1a 中的*核密度图*（kernel density plot）是一条平滑的逼近密度函数的线。

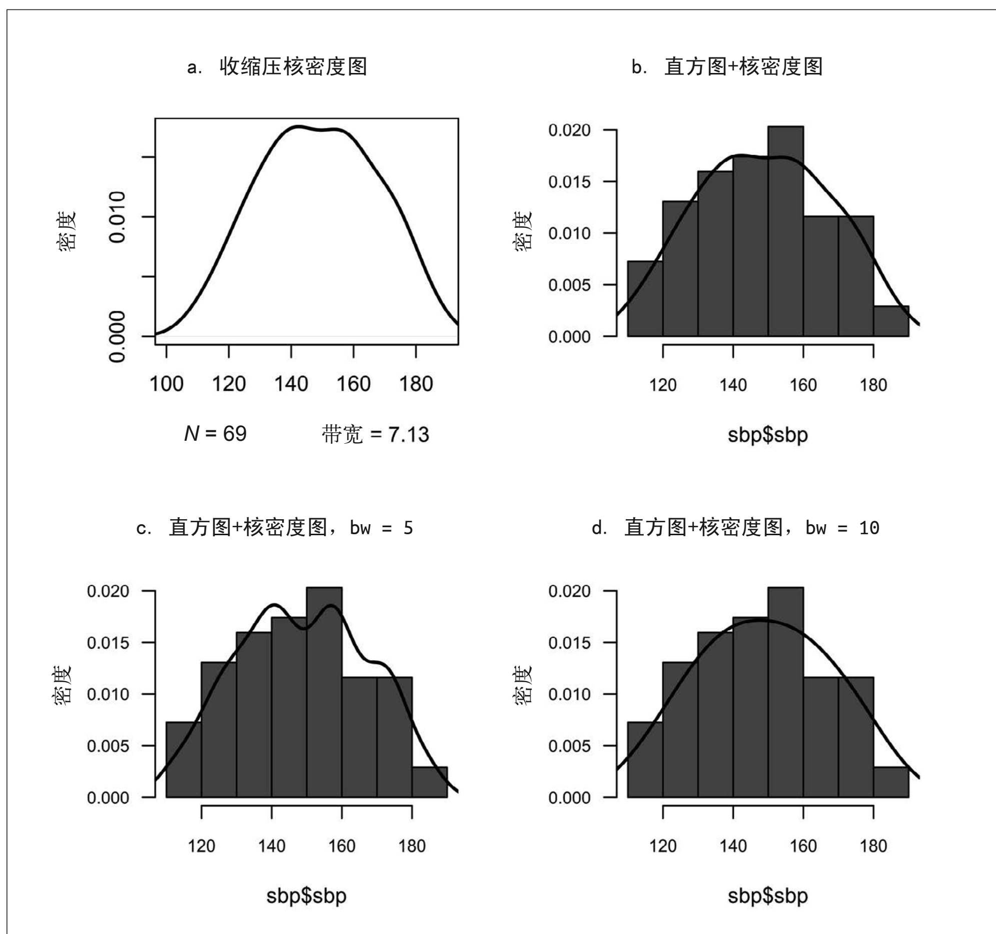


图 8-1：血压的核密度及加在血压直方图上的核密度

图 8-1b 是由 R 的 `density()` 函数生成的叠加在直方图上的密度估计。可以看到，密度曲线有时会高于直方图，有时会低于直方图。想象一下，取几个相邻组的数值的加权平均值，并用连接这些平均值的平滑线代替直方图的值。这就是一种平滑（smoothing）。图 8-1 中几个例子的脚本如下：

```
# 图8-1的脚本;有4张图
library(multcomp)
par(mfrow = c(2,2), cex.main =.9)

# 图8-1a
eq = density(sbp$sbp) # 估量sbp的密度曲线
plot(eq, xlim = c(100,190),
  main = "a. Systolic Blood Pressure Density Plot",
  lwd = 2) #  绘制估量
```

```
# 图8-1b
# 使用直方图来估量密度
hist (sbp$sbp,
  main = "b. Histogram + Kernel Density", col = "maroon",
  las = 1, cex.axis = .8, freq = F) # freq=F: 概率密度
lines(eq,lwd = 2) # 在已存在的直方图上绘制密度曲线

# 图8-1c
eq2 = density(sbp$sbp, bw = 5)
hist (sbp$sbp,
  main = "c. Histogram + Kernel Density, bw = 5",
  col = "maroon", las = 1, cex.axis = .8,
  freq = F) # freq=F: 概率密度
lines(eq2,lwd = 2) # 在已存在的直方图上绘制密度曲线

# 图8-1d
eq3 = density(sbp$sbp, bw = 10)
hist (sbp$sbp,
  main = "d. Histogram + Kernel Density, bw = 10",
  col = "maroon", las = 1,
  cex.axis = .8, freq = F) # freq=F: 概率密度
lines(eq3, lwd = 2) # 在已存在的直方图上绘制密度曲线
```

图 8-1a 是核密度图。核（kernel）指的是用来估计组成图的点的方法。在图 8-1b 中，采用 `lines()` 命令生成一个新图，叠加在现有的直方图之上。以这种方法合并两张图是非常有用的，并且我们经常会使用这种方法的变种。本例中，我们查看了概括单一分布的两种不同方法。

在图上添加曲线

`lines()` 函数是又一个用来在当前图上添加新信息的工具。（还记得 `abline()`、`text()`、`axis()` 等函数吗？）`abline()` 只能绘制直线，而 `lines()` 几乎可以绘制任意形状的线。`lines()` 接受的参数可以是包含定义线的点的向量，例如图 8-1 的脚本中的向量 `eq`；也可以是一对用来画线的变量 `x` 和 `y`。更多参数可以是 `par()` 中的参数。欲知更多信息，请输入 **`?lines`** 查询。

图 8-1a 在 x 轴上显示了默认标签，代表样本的大小 N 和带宽（bandwidth）。带宽表示图的分散程度。图 8-1b、图 8-1c 和图 8-1d 展示了改变带宽的效果。图 8-1b 中是默认的带宽，与图 8-1a 相同。注意，它符合直方图大体的形状。它改变了 3 次方向，即线有 3 个（非常小的）弯曲。我们很可能会得出这样的结论：线和直方图契合得很好。图 8-1c 展示了较小的带宽，它和直方图契合得更紧并改变了 5 次方向。图 8-1d 展示了较大的带宽，结果为一条平坦的线，仅弯曲一次。

8.1.1 选择带宽

应该使用多大的带宽？这并不总是很容易决定，并且给出一个准确的答案也超出了本书的

范围。或许使带宽非常小，即创建一条非常接近直方图的线是最好的方式。然而，请记住，直方图基于数据的一个样本。如果我们选取另一个样本，甚至是从 sbp 数据集中 69 个血压中选取的一个样本，直方图中的波峰和波谷也会有所不同，甚至有很大不同！越努力使密度曲线精确，就越不可能很好地契合新样本的直方图。作出过分详细的估计被称作过拟合（overfitting），这可能会导致令人尴尬的错误重新出现在新数据中。另一方面，如果使带宽变大，这样可能刚好适合大多数样本，却不会捕捉到很多细节。在许多情况下，你可能会发现利用不同带宽进行试错可以带来启发。一般来说，非常密集的数据（即大量数据）可以使用较小的带宽，而对于非常稀疏的数据建议使用更大的带宽。许多 R 包提供密度估计和寻找合适带宽的方法。例如，ASH 和 KernSmooth 对于大型数据集来说运行得特别快，而 np 虽提供基于数据的带宽计算，但相对缓慢。

8.1.2 比较两个或多个密度图

有时也需要比较两个或更多密度图。例如，你可能想比较两种分布的方式，看看它们是否有相似的形状和差异。考虑 1.8.2 节的 emissions 数据，代码如下：

```
> load("emiss.rda") # 加载来自第1章的emissions数据
> emissions  # 查看emissions数据

  Year  N_Amer CS_Amer Europe Eurasia Mid_East Africa
1 2004   16.2     2.4    7.9     8.5      7.1    1.1
2 2005   16.2     2.5    7.9     8.5      7.6    1.2
3 2006   15.9     2.5    7.9     8.7      7.7    1.1
4 2007   15.9     2.6    7.8     8.6      7.6    1.1
5 2008   15.4     2.6    7.7     8.9      7.9    1.2
6 2009   14.2     2.6    7.1     8.0      8.3    1.1
7 2010   14.5     2.7    7.2     8.4      8.4    1.1
 Asia_Oceania
1      2.7
2      2.9
3      3.1
4      3.2
5      3.3
6      3.5
7      3.6
```

假设需要比较欧洲和欧亚大陆的排放量。首先，我们可以首先为欧洲的碳排放数据绘制密度图，然后用 lines() 函数在生成的图上绘制欧亚大陆的数据。注意，下面生成图 8-2 的代码使用参数 xlim 和 ylim，以使欧洲密度图足够大，以免欧亚密度图超出图形。不要以为我有看似很棒的先见之明——我第一次尝试时没有扩展图的限制，结果弄得一团糟！有时试错是获得启发的唯一途径。生成图 8-2 的脚本如下：

```
# 图8-2的脚本
# 使用par()设置黑底白字的效果
load("emiss.rda")
par(bg = "black", col.lab = "white",
```

```
  col.axis = "white", bty="l",
  fg = "white", col.main = "white")
euro = density(emissions$Europe) # points on density(Euro)
ea = density(emissions$Eurasia) # points on density(Eurasia)

# 使用xlim和ylim,所以第二张图没有出界
plot(euro, xlim = c(6.9,9), ylim = c(0,2),
  main = "CO2 Emissions in Europe and Eurasia",
  col = "goldenrod1", lwd = 2)
lines(ea, xlim = c(6.9,9), ylim = c(0,2),
  lty = 2, lwd = 2, col = "cyan")
# lty=2 为虚线
# lwd = 2 为比默认线宽的线

legend("topleft",c("Europe","Eurasia"),
  text.col = c("goldenrod3","cyan"), bty = "n")
```

结果如图 8-2 所示。

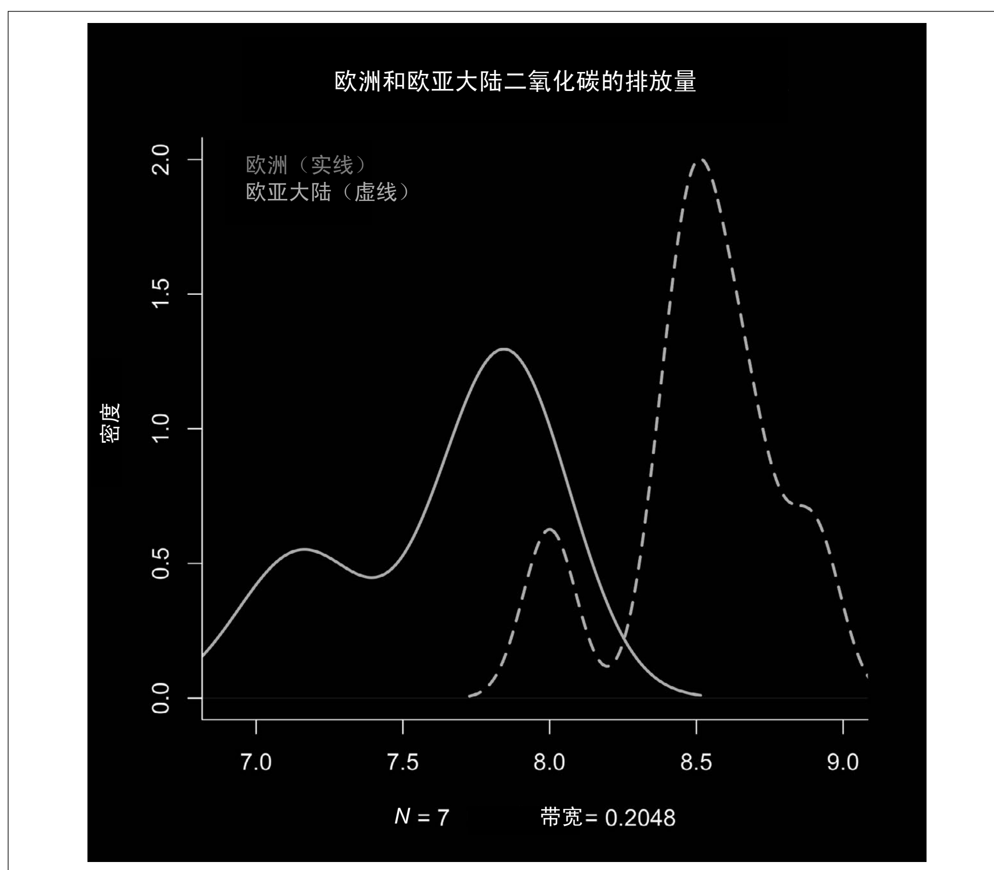


图 8-2：在同一轴上的两个密度图。 在第一张图上使用参数 xlim 和 ylim，让图的面积足够大以便包含第二张图，否则第二张图可能会出界

8.1.3 背景不是白色的

本书中大部分图的背景都是白色的，看起来干净、清晰。请注意，图 8-2 的背景是黑色的，这样做只是为了表示你想怎么做都行。如果运行生成图 8-2 的脚本时不用 `par()` 命令，通常会使用白色背景。再次运行脚本时包括 `par()` 命令，就会出现黑色背景。如果首先生成的是黑色背景，那么改为白色背景就不只是去掉 `par()` 那么简单了。一旦运行此命令，一些参数就会被改变，并且这些改变仍然有效。每个参数都需要重置，以作为另一个 `par()` 命令的参数，即变“黑”为“白”，反之亦然。顺便说一下，如果需要，可以输入不带参数的 **`par()`** 命令，看看哪些参数有效。

8.2 累积分布函数

密度图可以给人启示，但并不总是为我们提供真正需要的信息。尽管密度图能让我们感知水平轴上值的相对可能性，但我们常常更想知道数值小于等于 120、大于等于 135 或 120~140 的收缩压的可能性。*累积分布函数*（cumulative distribution function，CDF）图在 y 轴上展示一个分数小于等于 x 轴上数值的概率。

考虑一个遵循*正态分布*（normal distribution，也称“钟形曲线”）的数据分布。我们的例子来自一个*仿真*（simulation）或计算机模拟：从大量具有指定特征的数字中选出一个很大的样本。生成图 8-3 的代码如下，用 `rnorm()` 函数实现：

```
# 图8-3的代码
library(multcomp)
library(Hmisc)
par(mfrow = c(2,2), cex.main = .9, bg = "white")

# 从常规距离中抽取100 000个样本数据
# 其mean = 0 且 sd = 1
sam <- (rnorm(100000)) #mean = 0和sd = 1是默认值
plot(density(sam),
  main = "a. Density (sampling from Normal distribution)",
  col = "coral4") # 图 8-3a
polygon(density(sam), col = "coral4") # 曲线下的颜色区域

plot(ecdf(sam),
  main = "b. Cumulative distribution function of sample in
  Figure 8-3a", col = "turquoise")

plot(ecdf(sbp$sbp), main = "c. ecdf(sbp$sbp) - base R")
Ecdf(sbp$sbp,
  main="d. Ecdf(sbp$sbp) - Hmisc pack. + grid()",
  xlab = "sys blood pressure", col = "deepskyblue3")
grid(col = "gray70") # 为当前图添加灰色网格
```

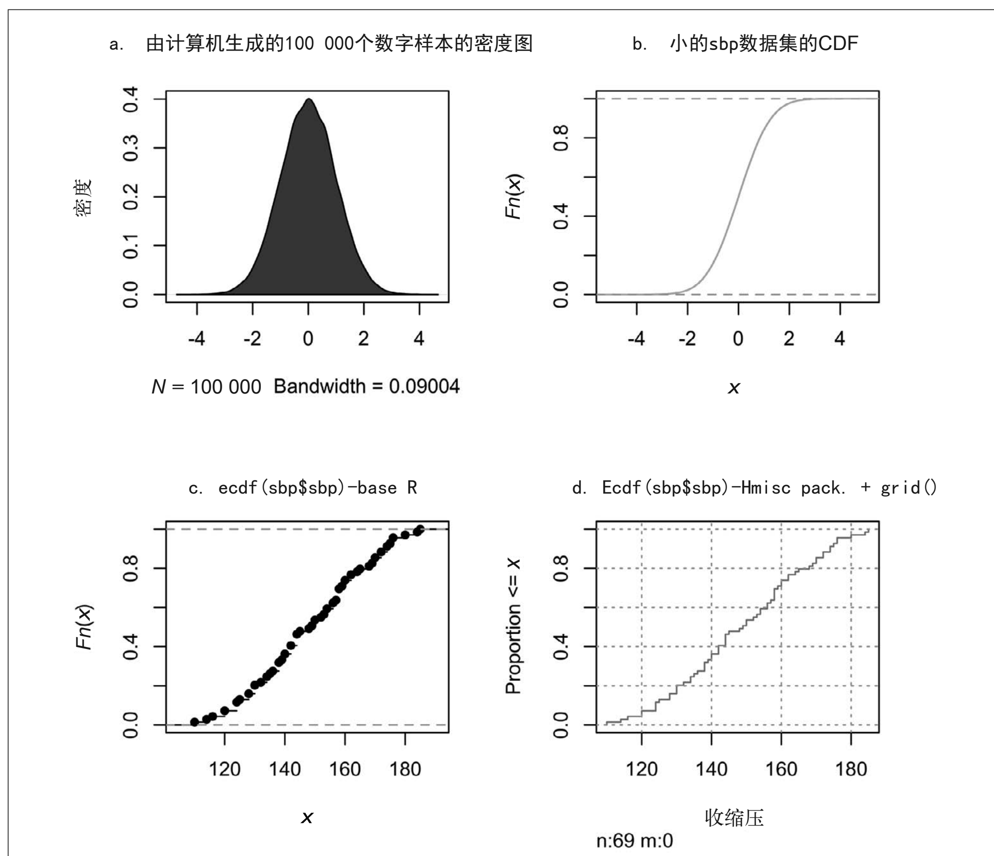


图 8-3：经验累积分布函数

图 8-3a 是由计算机生成的 100 000 个数字样本的密度图。图 8-3b 显示数据集的 CDF。由图可知，y 轴上大约一半的数字小于或等于 x 轴上的 0 值，而且几乎（但不完全）所有的数字都小于 2。图 8-3b 的 CDF 图是利用 R 的 `ecdf()` 函数生成的，`ecdf()` 是经验累积密度函数（empirical cumulative density function，CDF）。它是一条平滑的曲线，因为有很多数字并且分布是连续的。图 8-3c 展示了将 `ecdf()` 函数应用到小的 `sbp` 数据集时的情况。“曲线”以与图 8-3b 中相同的方式解释，但数据非常稀疏，以致图上有断点，这使此图缺乏吸引力且难以阅读。练习读图 8-3c。比大约一半血压都要高的血压值是多少？在 y 轴上找到 proportion = 0.5 的位置。x 坐标约为 150 或稍小一点，所以约 50% 的血压小于等于 150。

找到 CDF 的另一个方法使用 `Hmisc` 包中的 `ecdf()` 函数，其生成的结果如图 8-3d 所示。这是一个阶梯函数而不是光滑的曲线，但它比图 8-3c 更好看，且更易于阅读。这个函数有很多高级特性可用，比如为多个组生成图形的能力以及一些标签选项。还有其他生成 CDF 的方式，其中非常有趣的一个是 `ggplot2` 包中的 `stat_ecdf()` 函数，其生成的图如图 8-4 所示。它既好看又易于阅读，主要是因为有网格线。生成图 8-4 的代码如下：

```
# 图8-4的代码
library(ggplot2)
ggplot(sbp, aes(x=sbp)) + stat_ecdf()
```

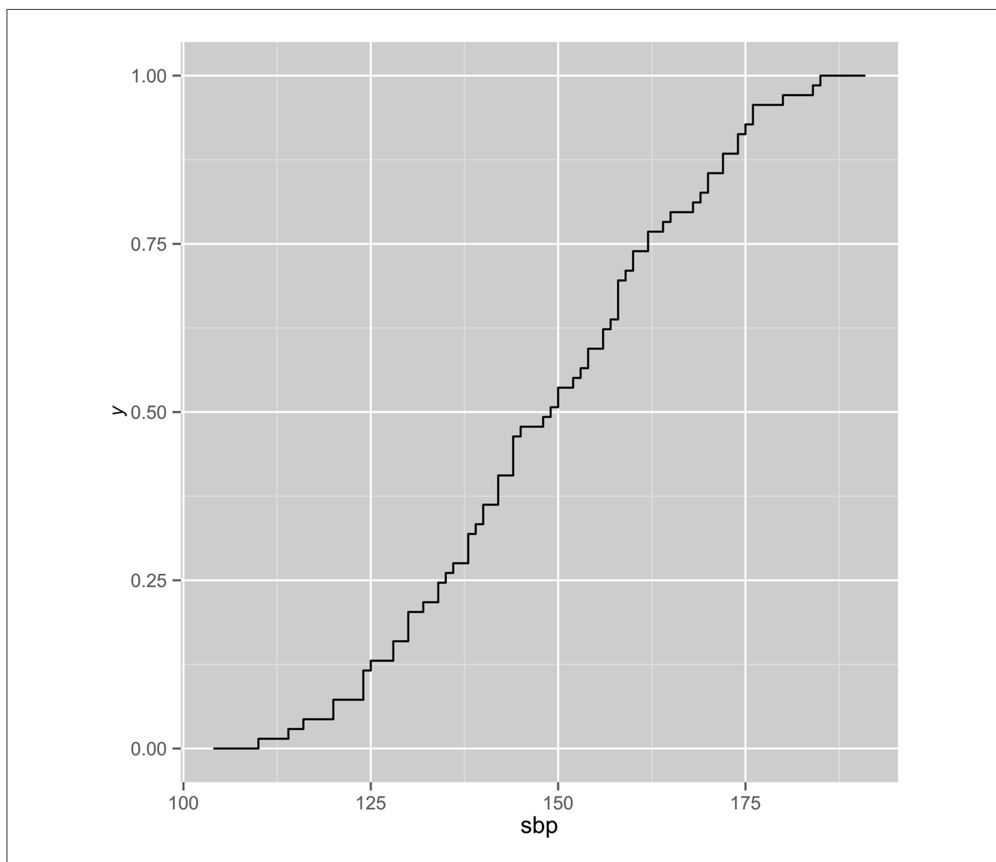


图 8-4：由 ggplot2 包生成的经验累积密度图，网格线使这张图易于阅读

8.2.1 练习8-1

继续进行图 8-1 的实验。选择各种带宽，并绘制基于直方图的密度图。使用一些接近图 8-1 的带宽和一些完全不同的带宽。你学到了什么？

8.2.2 练习8-2

基于图 8-4，随机选择一个收缩压为 125 或更低的人的概率是多少？大于等于 175 的概率呢？大于等于 125 但小于等于 175 的概率是多少呢？

第 9 章

条形图

9.1 基础条形图

让我们回顾第 7 章的 Salaries 数据集。图 7-5 展示了六张直方图，分别代表不同级别和性别的六组教师的工资分布。查看该数据的另一种有趣的方式是比较各组的教师总人数。让我们由简入繁，先看只代表了三个不同级别的教师总人数的图形。知道了每个级别的总人数，我们可以将它们输入一个向量，如下所示：

```
> ranknum = c(67,64,266)
```

然后，绘制 ranknum 向量的条形图（bar plot 或 bar chart）：

```
> barplot(ranknum)
```

如果总数未知，可以用 table() 函数把这些总数存在一个向量中，然后用 barplot() 操作该向量，代码如下：

```
# 为图9-1做准备
install.packages("car") # 若还没安装car
library(car)
attach(Salaries)
rankcount = table(rank) # 获取rank的计数并保存到向量
rankcount # 打印结果

rank
 AsstProf AssocProf      Prof
       67        64      266
```

如下代码中的 barplot() 函数以条的高度代表向量中的元素，在本例中即为各级别教师的总人数。这样图中就有三个竖条，前两个的高度几乎相等，第三个的高度约为其他两个的四倍：

```
# 图9-1a
barplot(rankcount, ylab = "Count", col = "skyblue",
  main = "Faculty by Rank", sub = "a. Number in each rank")
```

图 9-1a 展示了该条形图。

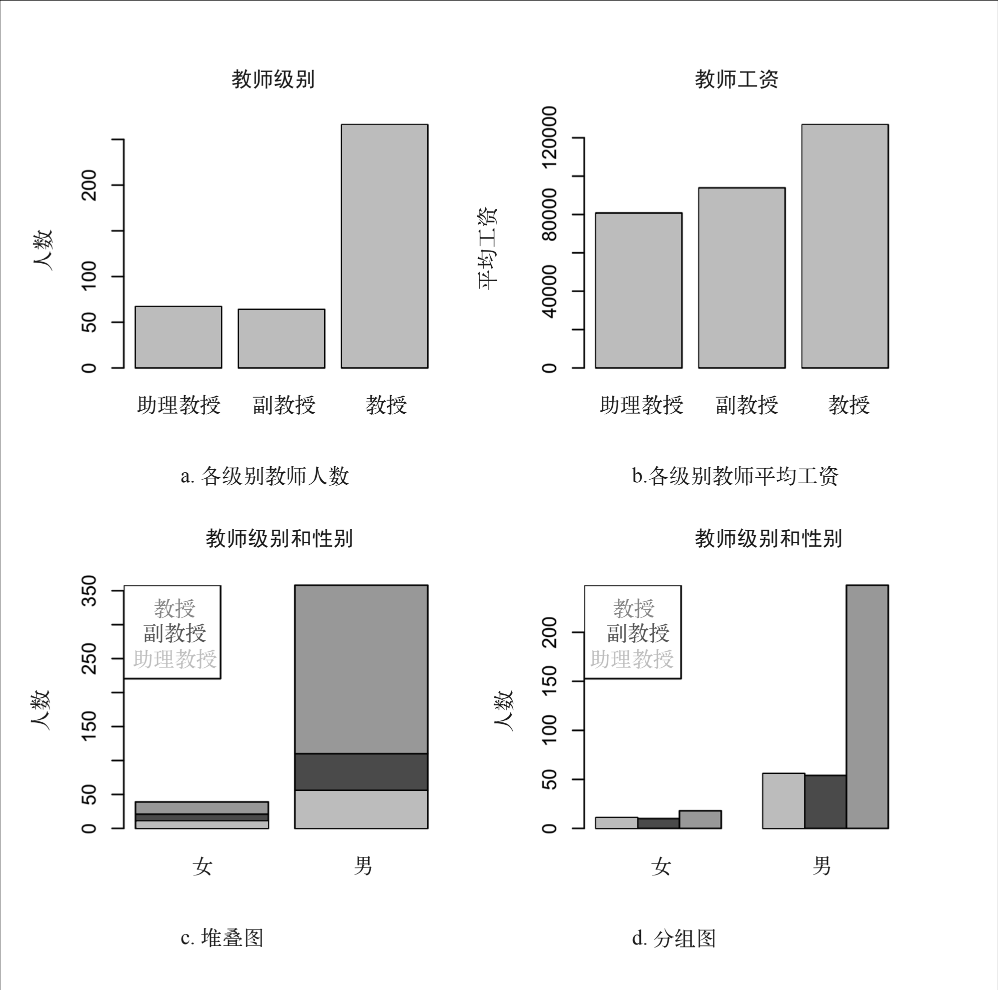


图 9-1：每一类中教授数量的条形图，每一类中平均工资的条形图。每类分别按级别和性别计数

注意，尽管条形图看起来有点像直方图，但二者完全不同。直方图的条是通过将沿着坐标轴不断增加（或减少）的定量变量分成多个部分来定义的。如果需要，你可以使用不同的

断点定义条。而条形图的条则是离散的，甚至是按类区分的，所以断点通常是固定的或者逻辑上不可移动的。条形图的条可以代表男性 / 女性、马 / 牛 / 猪、山 / 海滨、黄金 / 钻石 / 纸币或其他任何互斥且不是定量的分类。基于直方图拟合密度图是有意义的，但通常不会基于条形图进行拟合。

在图 9-1a 中，每一个条的高度代表相应级别教师的人数。虽然条形图通常用来显示总数，但条的高度可以代表任何数值，例如测量值、均值、税后收入等。图 9-1b 中的条形图就是一个例子，它展示了不同级别教师的平均工资。为了绘制这样一张图，首先必须使用 `aggregate()` 函数把平均工资放到一个向量中，然后调用 `barplot()` 来操作新创建的数据。表达式 `salary ~ rank` 表明对三个级别中的工资执行操作。`FUN = mean` 表明将找到均值，代码如下：

```
aver = aggregate(salary ~ rank, FUN = mean) # aver 是一个新的向量
aver      # 看看aver中有什么
       rank    salary
1  AsstProf  80775.99
2 AssocProf  93876.44
3      Prof 126772.11

# 图9-1b条形图的高度表示平均工资,名称为等级
barplot(aver$salary, ylab = "Average Salary",
  names.arg = aver$rank, col = "skyblue",
  main = "Faculty Salaries", sub = "b. Average salary by rank")
```

你可以通过修改条形图来展示两个变量的关系。绘制堆叠条形图（stacked bar plot）就是一种方法，如图 9-1c 所示。在这个示例中，分别代表男性教授和女性教授数量的两个条进行了细分，以展示各级别男教授和女教授的人数。为绘制此图，首先要创建一张表 `rank2`，将所有教授按级别和性别进行分组，代码如下：

```
rank2 = table(rank,sex)
rank2
          sex
rank       Female Male
  AsstProf       11   56
  AssocProf      10   54
  Prof           18  248
```

`rank2` 中的数据可以用 `barplot()` 呈现（见图 9-1c），代码如下：

```
# 图9-1c
barplot(rank2, ylab = "Count", names.arg = c("Female","Male"),
  main = "Faculty by Rank and Sex",
  col = c("skyblue","skyblue4","burlywood"),
  sub = "c. Stacked plot")
legend("topleft", c("Prof","Assoc","Asst"),
  text.col = c("burlywood","skyblue4","skyblue"))
```

堆叠条形图有时很难解释，因此你可能想考虑其他的选择。代表每个级别和性别的组合的

各种矩形，可以分别变成单独的条，并分组。在本例中，你可以把同一性别的所有条放在一起实现分组。新的结果如图 9-1d 所示，绘制该图需要修改用来创建图 9-1c 的代码，即添加参数 beside = T，代码如下：

```
# 图9-1d
barplot(rank2, ylab = "Count", names.arg = c("Female","Male"),
  main = "Faculty by Rank and Sex",
  col = c("skyblue","skyblue4","burlywood"),
  sub = "d. Grouped plot", beside = T)
legend("topleft", c("Prof","Assoc","Asst"),
  text.col = c("burlywood","skyblue4","skyblue"))
```

注意，生成图 9-1c 和图 9-1d 的代码添加了 legend() 函数，目的是给图添加额外的文本，以便解释各种颜色和符号的意思。图例可能是必要的，也可能无关紧要，有时甚至是有害的，这一点取决于具体情况。图例可能会导致混乱，因此图例的取舍非常重要。如果确定需要图例，那么下一步是决定把图例放在哪里效果最好。通常，图例应当放在图中离重要图形相对较远的地方。注意，在图 9-1c 和图 9-1d 中，图例都在左上角，这是通过参数 "topleft" 实现的。图例中的名称和 rank 的值相对应，并且颜色向量（col=）与 barplot() 命令中的颜色向量一致。

9.2 脊柱图

堆叠条形图读起来有点困难，你可以使用一种被称为*脊柱图* [spine plot，也被称为 spinogram 或者*比例堆叠条形图*（proportional stacked bar graph）] 的变种来改进图 9-1c 中的堆叠条形图。在脊柱图中，六个矩形的面积分别与所在组合的教授人数成比例，就像在堆叠图中那样。而在条形图中，两个条的宽度相同，因此高度是特定性别 / 级别组合中人数的唯一指标。这导致部分条的高度比较小，以至于很难与其他条进行比较。脊柱图采取了另一种方法，即两个条的高度相同，但宽度不同。右边的刻度范围涵盖了 0~1 的区间，易于估计给定条内级别的比例。

比较图 9-2 中的脊柱图和图 9-1c 中的堆叠条形图。哪一个更容易理解？

生成图 9-2 的代码如下：

```
# 图9-2的脚本
rank3 = table(sex, rank)
rank3
        rank
sex      AsstProf  AssocProf  Prof
  Female       11         10    18
  Male         56         54   248

spineplot(rank3, col = c("skyblue","skyblue4","burlywood"),
  main = "Faculty by Sex and Rank")
```

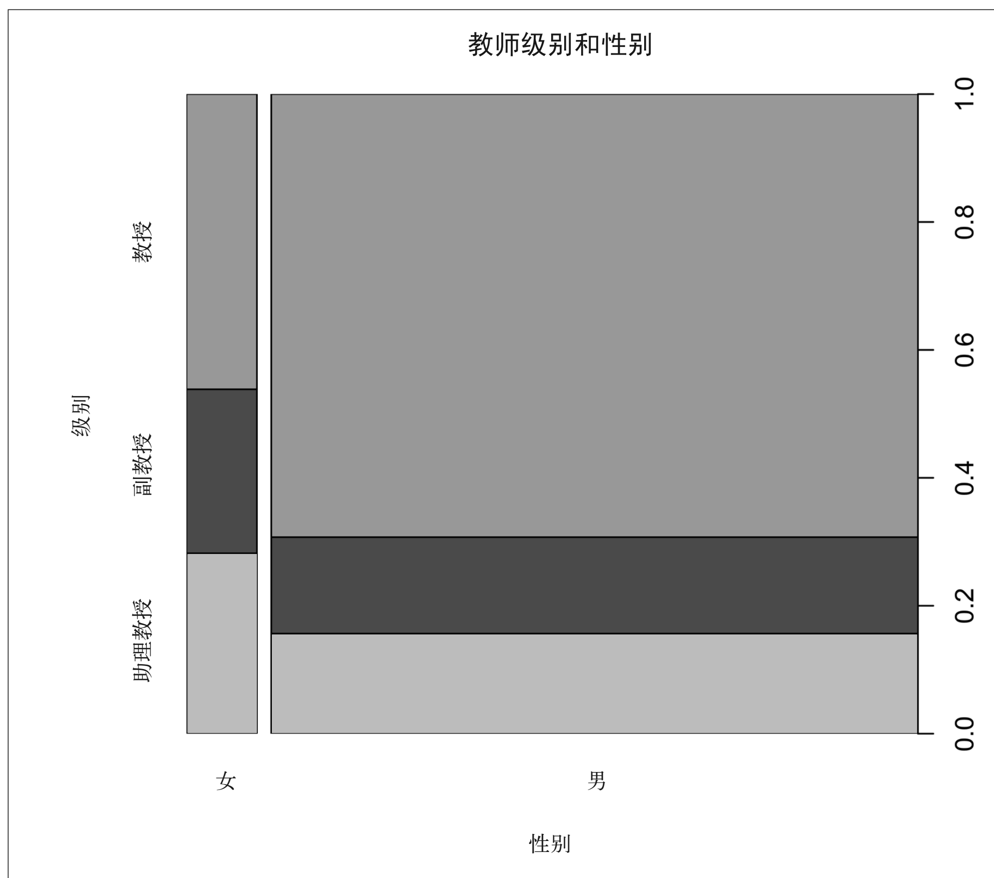


图 9-2：脊柱图。将其与图 9-1c 对比，哪一个更易于理解？

9.3 条形图的间距和方向

图中条的间距和方向对于传达信息十分重要。思考比较六种性别与级别组合的工资问题。各组平均工资有几种不同的呈现方法，脚本如下：

```
# 图9-3的脚本
library(car) # 图9-3
attach(Salaries)
par(mfrow = c(2, 2))
grp.sal = aggregate(
  salary ~ sex * rank, FUN = mean) # 每组的均值

# 标签多次重复使用,可以用命令行输入变量名称
rankname = c(" Asst", " ", " Assoc", " ", " Prof", "")
sexcol = c("blue", "maroon")
sexlab = c("Female", "Male")
```

```
# 图9-3a
barplot(grp.sal$salary, ylab = "average salary",
  names.arg = rankname, col = sexcol,
  main = "Faculty Salaries",
  sub = "a. Default spacing between bars")
legend("topleft", sexlab, text.col = sexcol,
  text.font = 2, title = "Sex",
  title.col = "black", cex = 0.8)

# 图9-3b
barplot(grp.sal$salary, ylab = "average salary",
  names.arg = rankname, col = sexcol,
  main = "Faculty Salaries", space = 1.5,
  sub = "b. Wide space between, space = 1.5")
legend("topleft", sexlab, text.col = sexcol, text.font = 2,
  bty = "n")

# 图9-3c
barplot(grp.sal$salary, ylab = "average salary",
  names.arg = rankname, col = sexcol,
  main = "Faculty Salaries", space = c(1, 0, 1, 0, 1, 0),
  sub = "c. Same rank together, space = c(1,0,1,0,1,0)")
legend("topleft", sexlab, text.col = sexcol,
  text.font = 2, bty = "n")

# 图9-3d
barplot(grp.sal$salary, ylab = "average salary", col = sexcol,
  main = "Faculty Salaries", space = c(1, 0, 1, 0, 1, 0),
  horiz = T, sub = "d. Horizontal version of c. horiz=T",
  names.arg = rankname,
  cex.names = 0.8, las = 1)
legend("bottomright", sexlab, text.col = sexcol,
  text.font = 2, bty = "n")

detach(Salaries)
```

首先，使用 aggregate() 函数得到各组的平均工资向量 grp.mean。表达式 salary ~ sex * rank 表明对六个组中的 salary 分别执行操作。FUN = mean 表明操作将找出均值。我们将绘制几张条形图，展示条间不同的间距，并且把垂直的条变为水平的条。这样的变化可以让我们从另一个角度看图。

下一步是定义字符向量 rankname、sexcol 和 sexlab。你可以采用相对简短的向量名，而不必在每次调用 barplot() 函数时输入字符串。这不是硬性规定，但这样做确实更方便。

图 9-3 显示了生成的四张条形图。

每个绘图命令后面跟着一个 legend() 命令，在生成的条形图上添加图例。你可以在控制台中逐条输入所有的命令行，但如果希望在屏幕或页面上查看结果，那么把一组命令写入脚本通常更方便。这样，如果你犯了一个错误，仅更正这个错误并再次运行整批命令即可，不需要重新输入所有命令。

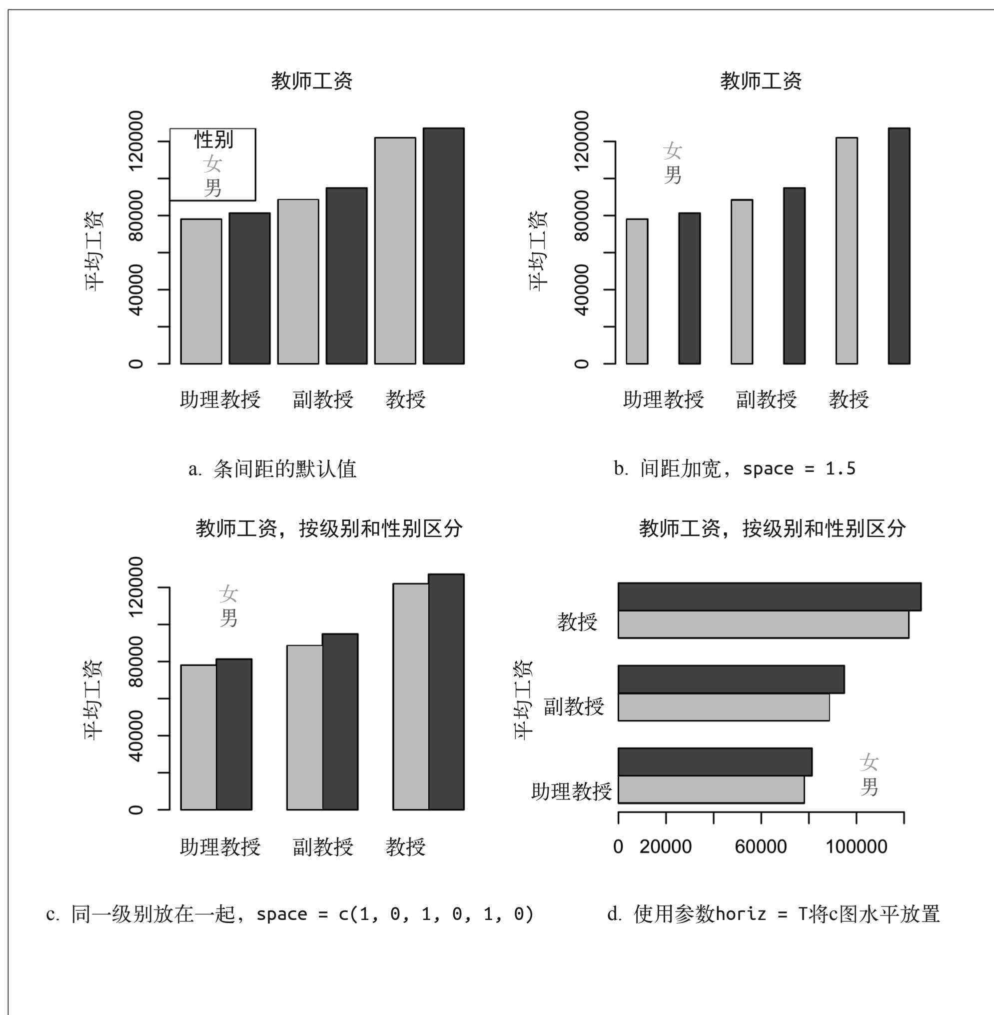


图 9-3：一个图的四个变形，把级别和性别并列放置

这四张条形图在分组、颜色、标签和图例方面非常相似。最重要的区别是条的间距，另外，最后一张图的方向和其他图不同。第一张图（见图 9-3a）中没有出现参数 `space`，因此使用了默认值绘图。图 9-3b 使用了 `space = 1.5`，所以条间隔很宽，即条间的宽度是条的 1.5 倍。最后两张条形图中，各级别中代表男性和女性的条紧邻，而不同的级别是分开的。这是通过使用参数 `space = c(1, 0, 1, 0, 1, 0)` 实现的，它告诉 R：在第一个条前应该有一个大小为 1 的间隔，在第二个条前应该有一个为 0 的间隔，以此类推。

第一张图的 `legend()` 命令生成了一个有标题和边框的传统图例。本例中，这些元素并不是必要的，因此其他的图去掉了参数 `title`，并添加了 `bty = "n"`，后者会删除边框。

比较图 9-3 中的四张条形图。注意，图 9-3a 中的图例使图看起来有点杂乱。其他图直接把注意力吸引到了条上。如果仔细研究前两张图，会发现在每一个级别中女性比男性的平均工资低，而图 9-3b 中各条间较宽的间隔使这一点不太明显。最后两幅图的差异就显而易见了。这就表明，正如应该小心选择词汇来清楚表达你的意思，你也应该谨慎选择图形，尽量凸显要点。

9.3.1 练习9-1

图 9-3d 中的图例有点奇怪。对于前三张图，图例刚好合适，但当图由纵向变为横向时，图例的顺序也改变了。为什么？试着去解决这个问题。

9.3.2 练习9-2

这个练习具有挑战性，但有助于你自测对 R 的掌握程度。请基于本章中用到的 `Salaries` 数据集，尝试重新生成图 9-4，这次使用 `epicalc` 包中的 `pyramid()` 函数来实现。结果是条形图还是直方图呢？

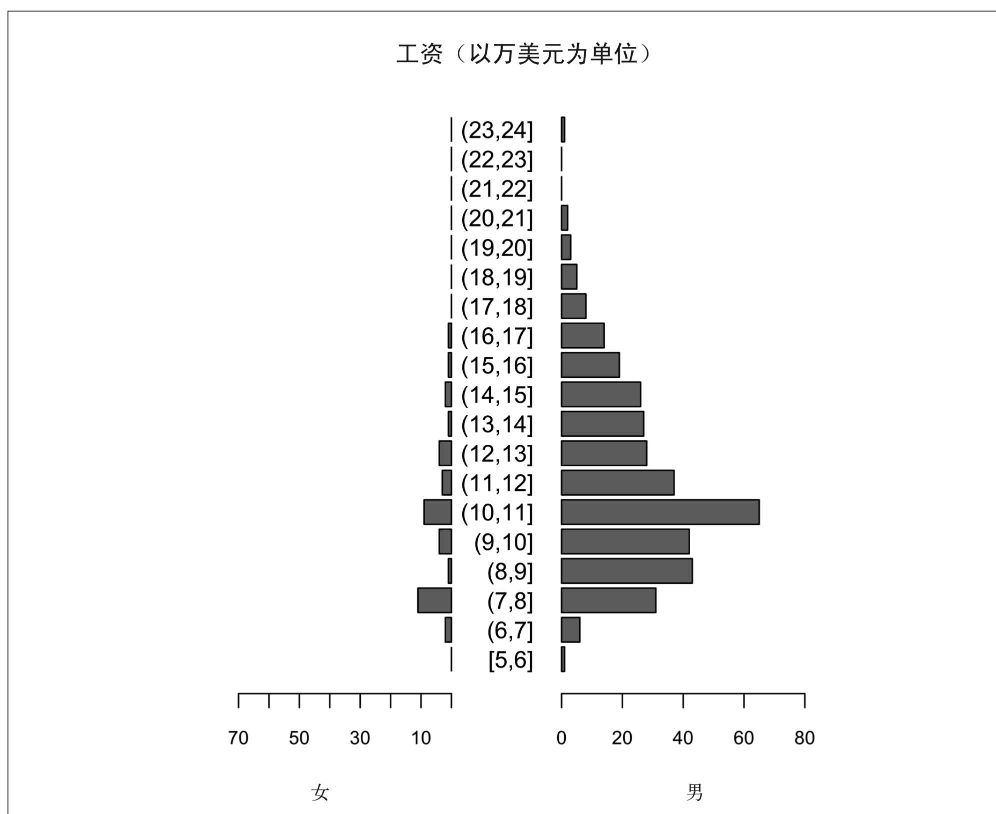


图 9-4：不同性别的工资

第10章

饼图

10.1　普通饼图

饼图（pie chart）是最常见的一类图。你一定已经见过无数饼图了。饼图在有些领域的应用已经深入人心，投资组合领域就是其中之一。投资顾问建议客户将持有的财富以指定的金量投入某些特定类别的项目，这样的建议通常以饼图的形式展现。基金经理也以类似的方式向客户报告他们（在一个时间点）的持股。考虑下面分配给“部门”的投资组合（顺便说一下，这不是建议）：

- 国内股票（domestic stocks）——30 %
- 国外股票（foreign stocks）——25 %
- 债券（bonds）——28 %
- 黄金 / 贵金属（gold/precious metals）——10 %
- 现金等价物（cash equivalents）——7 %

我们可以创建一个百分比的向量，并使用 `pie()` 函数来生成期望的图，代码如下所示：

```
# 图10-1的脚本
par(mfrow = c(2,2))

allocation = c(30,25,28,10,7)        # 投资分配

# 可以重复使用sector和sectcol;无需重新输入
sector = c("Stock","For'n'","Bonds",
           "Gold","Cash") # 名称适合页面显示
sectcol = c("burlywood","turquoise","firebrick",
            "gold3","green4")
```

```
# 图10-1 左上角
pie(allocation, labels = sector, main = "pie, default colors")

# 图10-1 右上角
pie(allocation, labels = sector, col = sectcol,
  main = "pie, choose colors")

# 图10-1 左下角
install.packages("plotrix", dependencies = TRUE)
library(plotrix) # 首先,必须已经安装了plotrix
pie3D(allocation, labels = sector, col = sectcol, explode = .1,
  labelcex = .95, labelrad = 1.3, main = "pie3D")
# explode将块分开,labelrad是标签离开边缘

# 图10-1 右下角
barplot(allocation, names.arg = sector, col = sectcol,
  main = "barplot")
```

结果如图 10-1 所示。

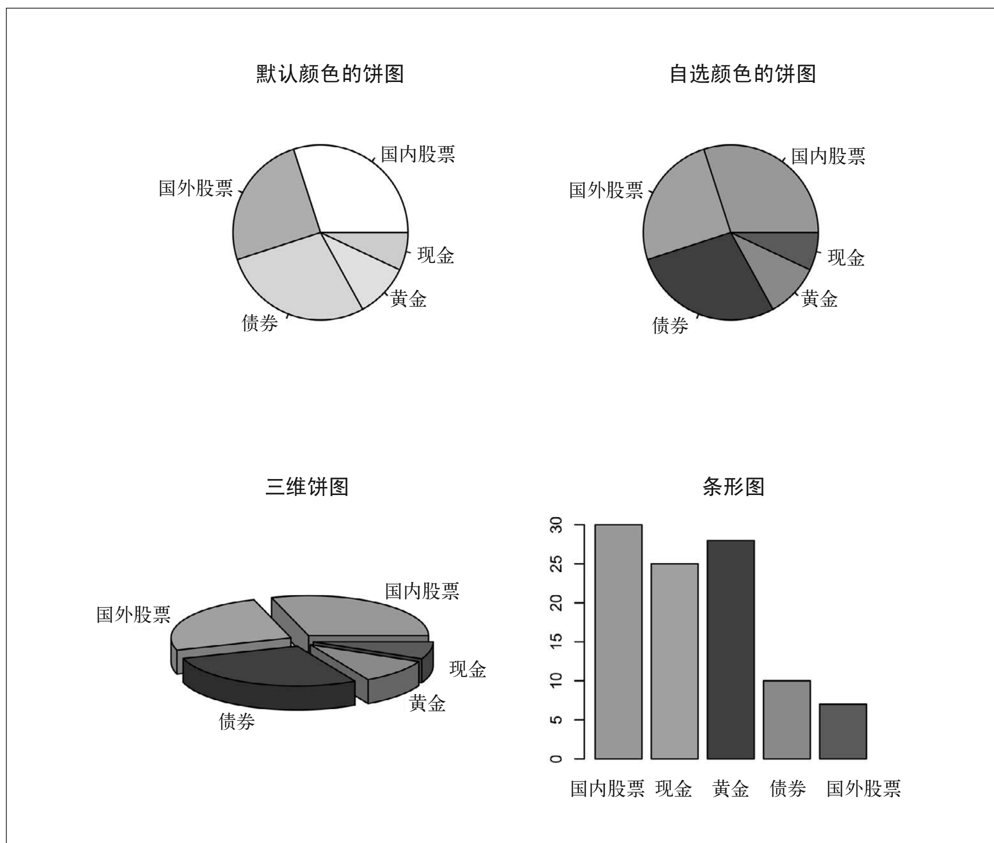


图 10-1：饼图和使用相同数据生成的条形图。注意，使用条形图比较分组的大小要容易得多

图 10-1 展示了三张饼图：第一张使用默认颜色，通过 `pie()` 函数创建；第二张使用向量 `sectcol` 中的颜色，通过 `pie()` 函数创建；第三张是三维视图（看起来很棒），通过 `pie3d()` 函数创建。还有一张条形图作为对照。注意，在饼图中，最大的三类看起来大小相同。较小的类别（黄金和现金）似乎也是相等的。然而，用完全相同的数据生成的条形图清楚地展示了差异。因此，统计学家对饼图评价不高，这没什么好惊讶的。

尽管有不足之处，有时饼图也很实用。当类别很少并且差异很明显时，这种图可能更受欢迎。当你想强调整体中由一个切片代表的某一部分时，饼图可以很好地满足需求。你可以以多种方式组织数据，从而绘制饼图。本章的练习可以帮助你加深理解。

10.2　扇形图

饼图的可替代方案是*扇形图*（fan plot），如图 10-2 所示。

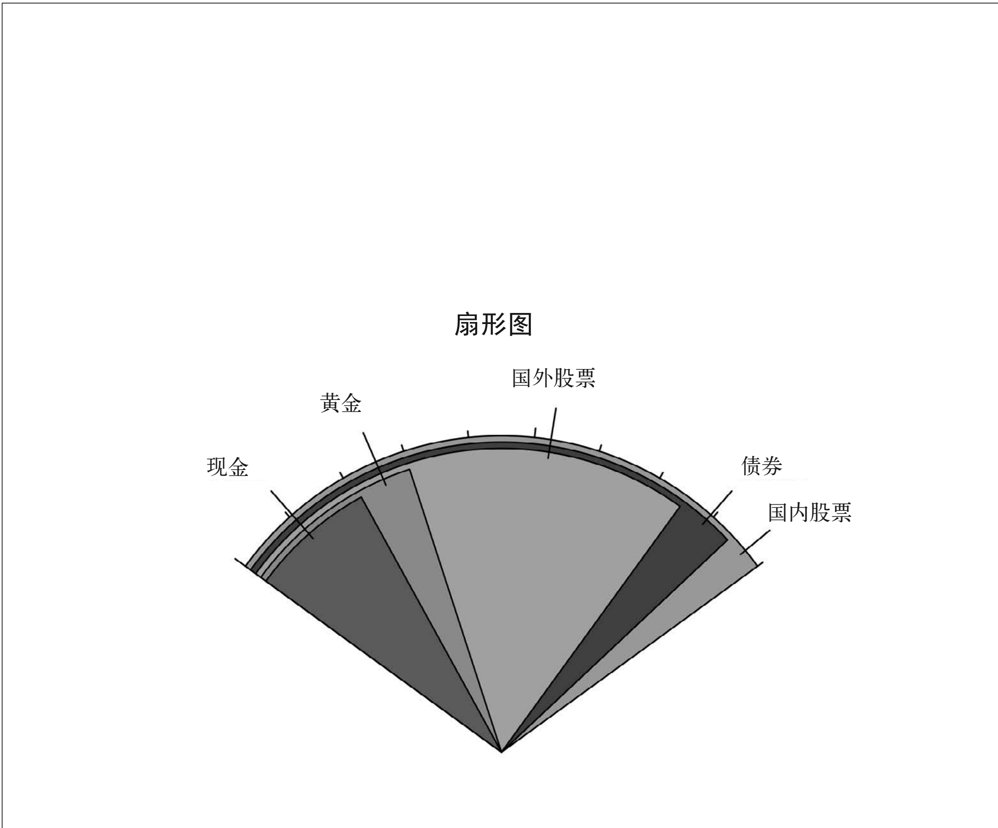


图 10-2：扇形图

这种类型的图看起来有点像饼图，但弥补了饼图最严重的缺陷。创建图 10-2 的代码如下：

```
# 图10-2
library(plotrix)
allocation = c(30,25,28,10,7)  # 投资分配
# 可以重复使用sector & sectcol;无需重新输入
sector = c("Stock","For'n","Bonds","Gold","Cash")
sectcol = c("burlywood","turquoise","firebrick","gold3",
  "green4")

fan.plot(allocation, labels = sector, col = sectcol,
  ticks = 30, main = "Fan Plot")
```

图 10-2 中的扇形图使用的标签与图 10-1 中的饼图相似，二者颜色相同。这看似有点混乱，因为图中楔形的大小不代表对应投资项目在投资组合中的比例，这个比例由不同颜色楔形的弧的长度表示。因此，我们从图上可以看出，“国内股票”比例最大，“债券”次之，以此类推。理解该图的另一种方法是，想象饼图上最大的切片被放在底部，第二大的切片放在其上，以此类推。最大切片的可见部分显示了与第二大切片的差额。同理，你也很容易看到第二大切片比下第三大切片大多少。如果清楚图的原理，这个思路是有帮助的，但如果你用它来展示自己的数据，那么需要仔细地解释该图。即便如此，有些人仍然难以理解这类图，而且他们会得出错误的结论，比如认为“国外股票”是图 10-2 中最大的。扇形图是非常聪明的设计，但长期的饼图经验会对扇形图的使用构成障碍。你要小心这一点。

10.2.1　练习10-1

请为在克里米亚战争中英国士兵死亡的原因绘制一张饼图。你可以在 `HistData` 包中的 `Nightingale` 数据集里找到数据。你需要首先安装并加载该包。请注意数据集的构造：三个独立的变量代表三类死亡原因。你需要新创建一个有三个数字的向量：数字为每个变量的总和。代码如下：

```
install.packages("HistData")
library(HistData)
attach(Nightingale)
deaths = c(sum(Disease), sum(Wounds), sum(Other))
```

请解析这段代码。相比投资组合的例子，本例是否更好地使用了饼图？请给出答案并说明原因。

10.2.2　练习10-2

为 `Nimrod` 数据集中的 `medium` 绘制一张饼图。你需要创建一个向量，包含每个 `medium` 的频率。`table()` 命令可以实现该操作。请把这副图绘制得条理清晰、赏心悦目。

第11章

地毯图

地毯图

地毯图（rug）并不是真正独立的图。它是一种一维展示，可以添加到已有的图上，以说明其他类型图中没能呈现的信息。地毯图和带状图一样，沿坐标轴在各个点放置不同的符号，代表变量的值，只不过它使用短线来表示点。它可以放在图的底部（默认）或顶部（`side = 3`）。如果合适的话，例如搭配垂直的箱线图的情形，地毯图也可以放在图的左侧（`side = 2`）或右侧（`side = 4`）轴上。当两个观测值相等时，它们会叠加，导致线更暗，示例如下：

```
# 图11-1的脚本
library(multcomp)
par(mfrow = c(2,2))
stripchart(mtcars$drat,
  main="a. side = 3", method = "jitter",
  pch = 20, col = "sienna4")
rug(mtcars$drat, side = 3)

boxplot(mtcars$drat, main = "b. side = 2",
  col = "firebrick")
rug(mtcars$drat, col = "darkmagenta", side = 2)

hist(airquality$Ozone, main = "c. side = 1", col = "cyan4")
rug(airquality$Ozone, col = "cyan4")

boxplot(sbp$sbp,
  main = "d. side = 4", col = "darkorange3")
rug(sbp$sbp, side = 4, col = "cornsilk4")
```

结果如图 11-1 所示。

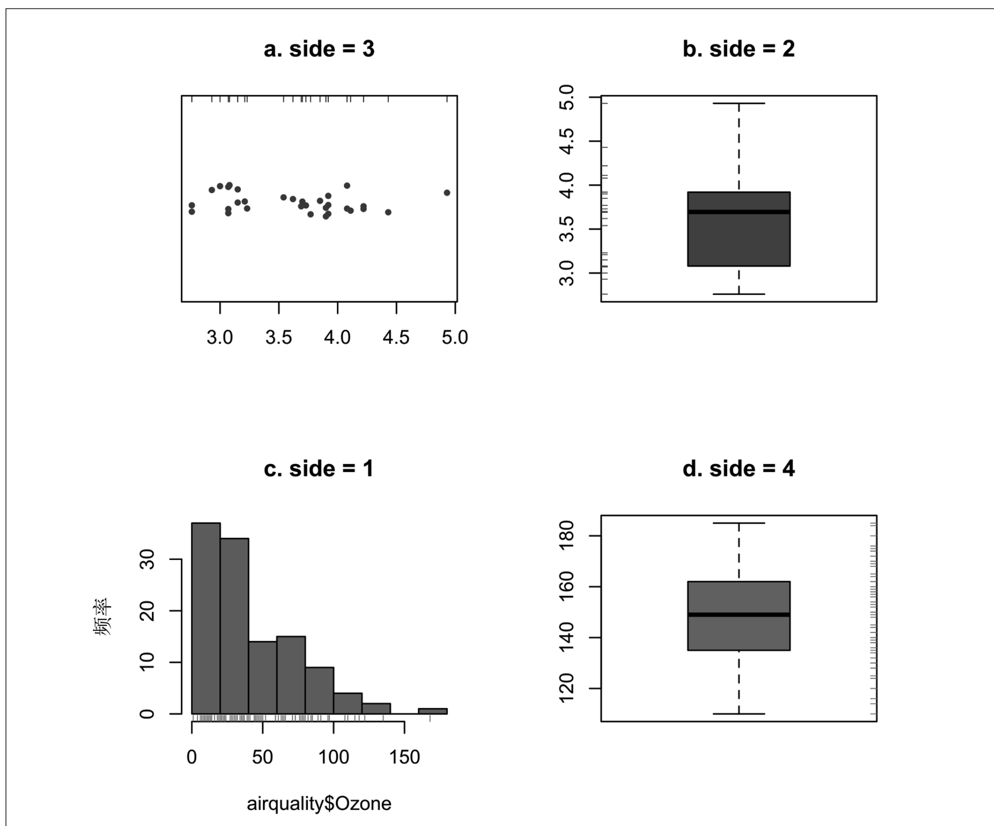


图 11-1：地毯图的应用

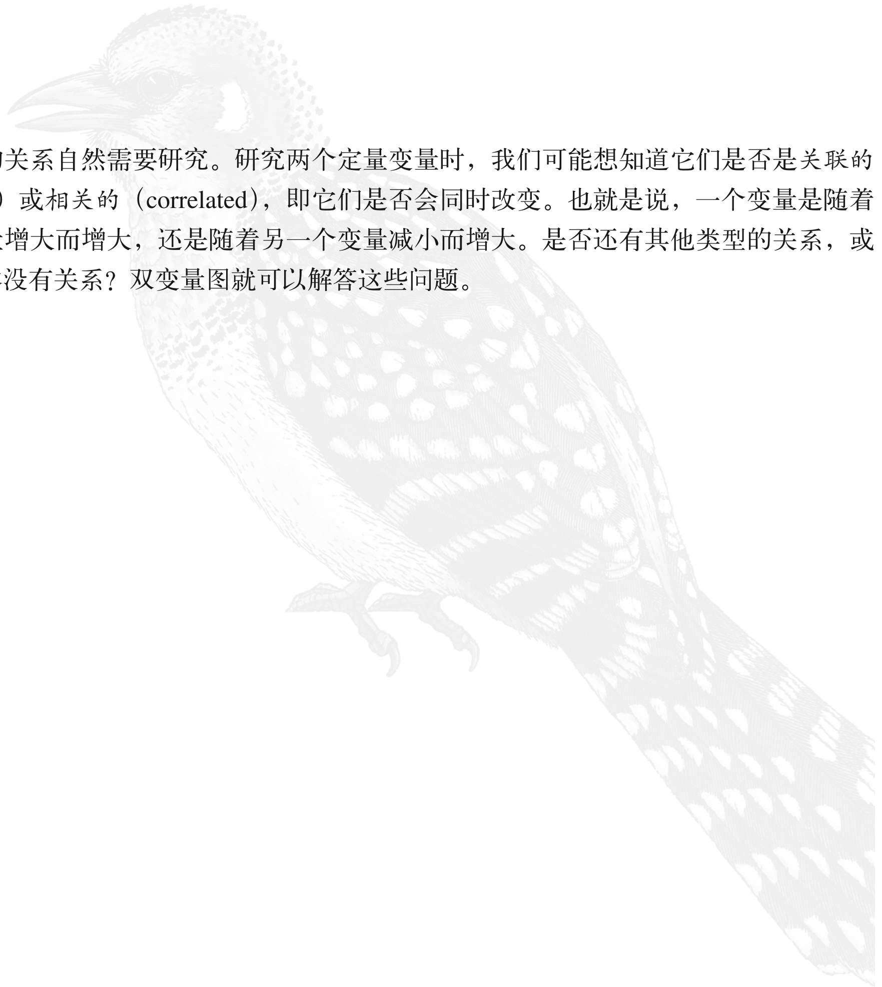
如图 11-1a 所示，添加地毯图本质上就像在一张图的底部或顶部加一个带状图。对于带状图，地毯图是多余的，没有什么帮助。但是，地毯图可以帮助其他类型的图展示可能丢失的信息。例如，在图 11-1b 中，箱形图呈偏态分布，但是仅凭箱线图我们不可能知道数据分布在几块中，而地毯图清晰地展示了这一点。图形上面的长细须，可能是几个分散的点或一个极值的结果。地毯图显示了所有点和它们的位置，包括一个极值和刚超过第三个四分位处聚集的几个点。将其与图 11-1d 对比，在图 11-1d 的整个范围内地毯几乎是等距的。图 11-1c 的地毯图再次将数据显示成块状。这说明改变箱子的大小可以改变直方图的形状。默认情况下，地毯图被放置在图的底部，但可以使用参数 `side = 3` 把它放在顶部。关于可用参数的更多信息，请输入 **`?rug`** 进行查询。地毯图有时很有帮助，但在其他情况下，它并没有显示出真正的优势。

练习11-1

给 `Nimrod` 数据集中 `time` 的密度图添加地毯图。给 `MathAchieve$SES` 的箱线图（见图 5-2）添加地毯图。哪张地毯图更实用？

第三部分

双变量图



变量之间的关系自然需要研究。研究两个定量变量时，我们可能想知道它们是否是关联的（associated）或相关的（correlated），即它们是否会同时改变。也就是说，一个变量是随着另一个变量增大而增大，还是随着另一个变量减小而增大。是否还有其他类型的关系，或者也许根本没有关系？双变量图就可以解答这些问题。

第12章

散点图和折线图

12.1　基础散点图

散点图（scatter plot）可能是最实用的图形化工具之一。这种十分常见的图可以方便地用于研究两个变量之间的关联关系。进一步来说，其他许多类型的图无非是基础散点图的变种。

让我们再次研讨 `trees` 数据集。记住，`head()` 函数打印出了前 6 行。输入 **`trees`** 可以查看整个数据集，代码如下：

```
> head(trees)
  Girth Height Volume
1   8.3     70   10.3
2   8.6     65   10.3
3   8.8     63   10.2
4  10.5     72   16.4
5  10.7     81   18.8
6  10.8     83   19.7
```

在散点图上可以看到，这 3 个变量之间有很强的关系。我们将使用 `plot()` 函数生成散点图，该函数的基本形式如下：

```
plot(x-variable, y-variable, arguments...)
```

下面的脚本生成了 `trees` 数据的几张散点图：

```
# 4小段脚本生成图12-4的4张图
attach(trees)
par(mfrow = c(2,2), cex = .7)
```

```
# 图12-1a:仅在图片上展示了两点
trees2 = trees[1:2,] # trees2是一个子集,只有前2棵树
                     # 见侧边
plot(trees2$Height, trees2$Girth,
  xlim = c(63,80),
  ylim = c(7.8,10),
  xlab = "Height",
  ylab = "Girth",
  main = "a. First two trees")

# text() 允许在图上添加注释
text(72,8.1,labels = "(Height = 70, Girth = 8.3)",
  xlim = c(61,80),
  ylim = c(8,22))
text(65,8.8, labels = "(65, 8.6)",
  xlim = c(62,89),
  ylim = c(8,22)

# 图12-1b:注意一个基础图只需要很少的代码
plot(Height, Girth, main = "b. All trees")

# 图12-1c,图符见表12-1
plot(Height, Girth,
  main = "c. Change plot character, add grid",
  pch = 20,
  col = "deepskyblue")
grid(col = "gray70")

# 图12-1d # abline 在图上添加线性回归线
plot(Height, Girth,
  main = "d. Add regression line", pch = 20,
  col = "deepskyblue")
abline(lm(Girth ~ Height),
  col = "dodgerblue4",
  lty = 1,
lwd = 2) # 覆盖上一个图的网格(col = "gray70")
detach(trees)
```

结果如图 12-1 所示。

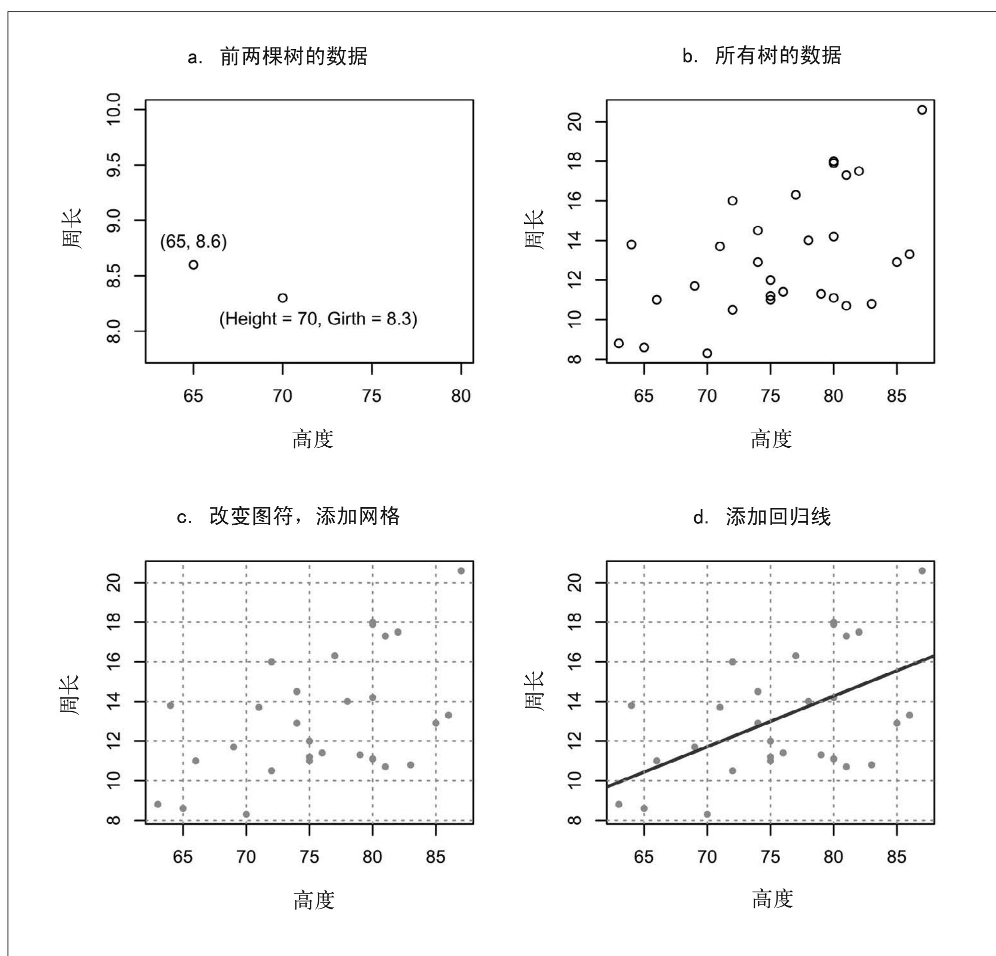


图 12-1：Height 和 Girth 的散点图

图 12-1 展示了散点图的使用。你可能已经忘记了多年前学的高中数学，所以图 12-1a 解释了怎样理解点（图中为小圆圈）。数据集中第一棵树的 Height（高度）为 70，Girth（周长）为 8.3。对应这些测量点，可以看到它们在图形上的位置。

图 12-1b 更进一步，在图上绘制了所有的点（即树的数据）。请注意，Height 和 Girth 之间似乎存在关系。随着 Height 变高，Girth 也会变长。这不是一种精确而严格的关系，但二者也不是随机搭配的。

图 12-1c 简单地改变了图符。这样做不仅让图更好看，也更易读。图中还引入了网格。grid() 函数为*活跃图*（active plot）添加了参考线。如果在创建图之后没有进一步发出命令，那么活跃图就是最近创建的一张图。默认情况下，在轴上的刻度线处绘制网格线。如果需要，可以改变这种方式，输入 **?grid** 查看如何实现。

我绘制的所有图都发生了什么

你可能想在一个单独的 R 会话中比较一系列图。如果仅依靠输入一条命令来生成一张图，那么建立新图的时候前一幅图通常会被删除，无法再次看到。不过，可以保持以前的图形窗口（多个窗口）始终打开。事实上，可以同时打开多达 63 个图形窗口。和 R 中大多数的任务一样，有多种方法可以做到这一点。顺便说一下，同时打开几个窗口可能是有帮助的，但不建议同时打开 63 个！

在所有平台上都可用的一种方法是，在输入生成下一张图的命令之前，输入 **`dev.new()`**。这样将创建一个空白的图形窗口，在该窗口中显示下一张图。之前创建的所有图形窗口都不会受到影响。然后，你可以重新研究已经绘制的任意一张图。点击你感兴趣的任意窗口即可，但如果有很多窗口，寻找想要的那个可能非常无聊。

如果你使用的是 Mac，更便捷的方法是在发出绘制新图的命令之前，打开“窗口”（Window）菜单，并点击“新石英设备窗口”（New Quartz Device Window）。同样，之前的图不会受到影响。打开“窗口”菜单，然后选择 Quartz2、Quartz5 等，可以很容易地从一张图切换到另一张图。

在基于 Windows 的计算机上，使用 `dev.new()` 创建一个新的图形窗口之后，打开“窗口”菜单，然后选择“R 图形设备 *n*”（R Graphics Device *n*），可以从一张图切换到另一张图。

另一种方法是创建一张图，并单击它所在的窗口。如果要将其保存，打开“文件”（File）菜单，单击“另存为”（Save As）即可。在 OS X 中，可以把图保存为 PDF 文件。在 Windows 系统中，可以选择将图保存为几种不同文件类型中的任何一种。（这两个平台上都有另一种方式可以将文件保存为各种格式，详见 2.1 节。）

如果你的文字处理器（或演示）程序允许，还有一种更为方便的方法，就是点击你想要的图，打开“编辑”（Edit）菜单，然后选择“复制”（Copy），再单击“粘贴”（Paste），把复制的图放到文字处理器中。遗憾的是，并不是所有的文字处理器程序都支持这种方式。在研究了所有的图之后，删除不要的图就好，剩下的图已经存在一个可以添加文本的文档中了。

通过使用 `abline()` 函数并作用于活跃图，图 12-1d 在点上添加了回归线。线性回归法是一种在观测数据中寻找“最佳拟合”直线的方法。相同横坐标下，拟合线上的点和被拟合线上的点的垂直距离就是“误差”。换句话说，误差显示了预测点的值距离线有多远。取所有误差的平方并求和，得到的参数可以衡量线的拟合程度。拟合线有无数条，可以放在图上的“最佳拟合”是“误差”平方和最小的线，即“最小平方”线。之前 `trees` 数据的脚本中，`abline()` 命令里有 `lm()` 函数，R 用此函数找到了拟合线。如果点更靠近线，我们可以推断，`Height` 和 `Girth` 之间的关系比在图 12-1d 中看到的更强。

回想直线方程的形式，Y 表示线上的一个点，如下所示：

```
Y = a + (b * X)
```

其中，a = 截距（线穿过 y 轴时的点），b = 斜率（Y 随 X 变化的比率）。

获取截距和斜率值的方法如下：

```
lm(Girth ~ Height)

Call:
lm(formula = Girth ~ Height)

Coefficients:
(Intercept)       Height
    -6.1884       0.2557
```

由公式可知，线是由方程确定的：

```
Girth = -6.1884 + (0.2557 * Height)
```

此外，使用以下命令可以得到与该模型相关的统计：

```
summary(lm(Girth ~ Height))
```

然而，对这一信息的解释超出了本书的范围。在其他情况下，我们可能已经在数据中看到了一种模式，该模式并不接近于一条直线，我们可能试图拟合一条曲线，或已经得出了两变量间没有关联的结论。有能力给图增加回归线固然很好，但如果不明就里，这就无异于玩火，所以要谨慎！

子　　集

在图 12-1a 中，trees2 是提取自 trees 的子集，即一个较小的数据集。对于比较部分与整体，或比较两个部分，子集是有帮助的。即使 R 提供了几种定义子集的方法，在脚本中使用的方法仍是优雅和经济的，只需要很少的输入。数据框 / 向量的名称后面的方括号中有两项内容：一个是关于行的表达式，另一个是关于列的表达式。

子集最简单的用途是查找元素。例如，查找第三行第二列的元素，代码如下：

```
> trees[3,2]
  [1] 63
```

或者用其中的数据创建一个新的向量：

```
> newrow = trees[3,2]
> newrow
  [1] 63
```

如果行表达式或列表达式为空，那么该子集包括整行或整列。如果想取整个第三行，可以使用如下代码：

```
> trees[3,]       # trees[-3,]  则表示第三行*以外*的所有内容

 Girth Height Volume
3  8.8     63   10.2
```

用符号 a:b 可以获取从 a 到 b 的元素。如果想取第四行到第六行，但只要第二列和第三列的所有值，那么实现代码如下：

```
> trees[4:6, 2:3]
  Height Volume
4     72   16.4
5     81   18.8
6     83   19.7
```

不连续的行或列可以用向量符号，通过数字和 / 或变量名来选择：

```
> trees[,c("Girth","Volume")] # trees[,c(1,3)] 做同样的事

  Girth Volume
1   8.3   10.3
2   8.6   10.3
3   8.8   10.2
4  10.5   16.4
5  10.7   18.8
...
```

使用如下方法可以删除有缺失值的任意行。

```
> mysubset = na.omit(airquality)
```

如果只需要选择那些具有某些特征的观测值，那么最好使用 subset() 函数，例如：

```
> subset(trees, Height > 70) # 仅有Height > 70的树

  Girth Height Volume
4   10.5    72   16.4
5   10.7    81   18.8
6   10.8    83   19.7
8   11.0    75   18.2
...
```

12.2 折线图

折线图（line chart，也称为 line graph 或 line plot）是一种特殊的散点图，它非常常见且非常实用。在这种类型的图中，任意两点具有不同的 *x* 值。此外，折线按照 *x* 值从小到大的顺序将所有点逐一连接起来。同一组坐标轴上可以显示两张或更多张折线图。用于散点图的 plot() 函数，也可生成折线图。图 12-2 呈现了几个实例。

我们用 HistData 包中的 Nightingale 数据集绘制图表，该数据集在练习 10-1 中首次出现。加载此包并查看数据，代码如下：

```
# 如果还没有安装HistData,那么必须安装,
# 否则,如下脚本无法运行
# install.packages("HistData", dep = T)
library(HistData)
attach(Nightingale)
head(Nightingale)   # head() 打印前6行

        Date   Month Year  Army Disease Wounds Other
1  1854-04-01    Apr 1854  8571       1      0     5
2  1854-05-01    May 1854 23333      12      0     9
3  1854-06-01    Jun 1854 28333      11      0     6
4  1854-07-01    Jul 1854 28722     359      0    23
5  1854-08-01    Aug 1854 30246     828      1    30
6  1854-09-01    Sep 1854 30290     788     81    70
  Disease.rate  Wounds.rate Other.rate
1        1.4           0.0        7.0
2        6.2           0.0        4.6
3        4.7           0.0        2.5
4      150.0           0.0        9.6
5      328.5           0.4       11.9
6      312.2          32.1       27.7
```

数据记录了英国士兵在克里米亚战争中每月的死亡人数。每一行代表一个月的数据，变量包括如月份、年份、军队规模，以及三种原因分别造成的死亡人数。每个日期（Date）因疾病（Disease）死亡的人数可以很容易地绘制成一张普通的散点图。你可以尝试一下。将这些点连接起来（第一个月到第二个月、第二个月到第三个月，以此类推），图会更有条理。为 plot() 添加参数 type = "b"，可以创建一个这样的基本折线图。你也可以添加参数 lty = "solid"，指定线的类型。（线也可以是点、虚线或其他类型；输入 **?par** 可查询更多信息。）下面的脚本可以生成图 12-2 的折线图：

```
# 图12-2,4张图
par(mfrow = c(2,2)) # 将4张图放在一个页面中
library(HistData)
attach(Nightingale)

# 图12-2a
plot(Date, Disease,
  type = "b",
  pch = 20,
  lty = "solid",
  main = "a. Line chart of Disease")

# 图12-2b
plot(Date,Disease,
  type = "l",
  lty = "solid",
  main = "b. Line chart, Disease, Wounds, Other")
lines(Date,Wounds,
  lty = "dashed",
  col = "red",
  lwd = 2)
lines(Date, Other,
```

```
    lty = "dotted",
    col = "navyblue",
    lwd = 2)

  # 图12-2c
  plot(Date, Disease,
    type = "h",
    lty = "solid",
    lwd = 20,
    main = "c. Change Disease to histogram",col="gray67",
      lend="butt")
  lines(Date,Wounds,
    lty = "solid",
    col = "red",
    lwd = 2)
  lines(Date, Other,
    lty = "dotted",
    col = "navyblue",
    lwd = 2)

  # 图12-2d
  plot(Date, Disease,
    type = "h",
    lty = "solid",
    lwd = 20,
    main = "d. Add legend, remove box",col="gray67",
    lend ="butt",bty="l")
  lines(Date, Wounds,
    lty = "solid",
    col = "red",
    lwd = 2)
  lines(Date, Other,
    lty = "dotted",
    col = "navyblue",
    lwd = 2)
  legend("topleft",
    c("Death from Disease","Death from Wounds","Other Deaths"),
    text.col = c("gray40", "red", "navyblue"),
    bty = "n",
    cex = .5)
  detach(Nightingale)
```

看一下图 12-2a，呈现该图的另一种方式是去掉点，使其成为一条完全相连的线，可以通过把 `type = "b"` 改为 `type = "l"` 来绘制这条线，结果如图 12-2b 所示。图 12-2b 应用 `lines()` 函数，在图上放置了两条额外的线，即因创伤（`Wounds`）和其他原因（`Other`）造成的死亡人数。

在战争中，死亡原因的差异是惊人的。对于许多战争来说，因疾病死亡的人数远远多于创伤和其他原因导致的死亡人数。尽管在图 12-2b 中这种结果是显而易见的，但我们可以通过一点简单的改动突出这一点。请看表 12-1 中参数 `type` 的选项，其中 `type = "h"` 为直方图，如图 12-2c 所示。在此有必要添加 `lend = "butt"`，使直方图条由圆角（line end 或 lend）变为方角。

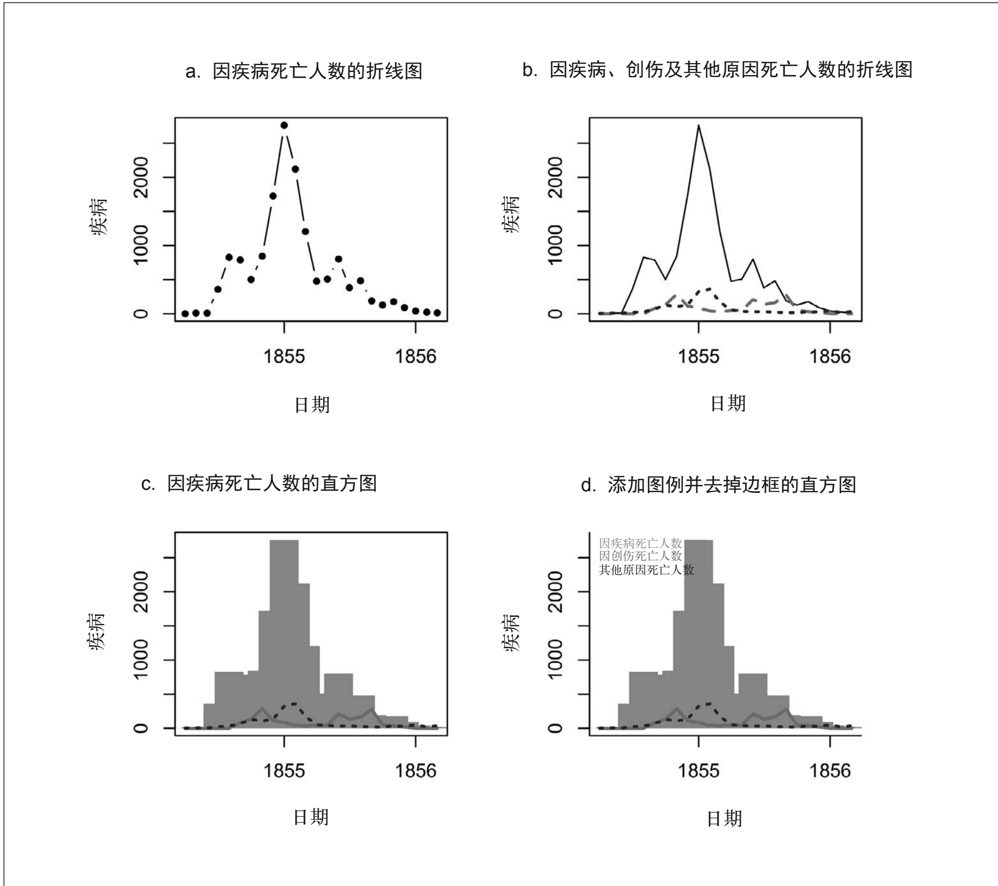


图 12-2：Nightingale 数据集中死亡原因的几种折线图的变形

图 12-2c 的呈现方式更有戏剧性，它在直方图中将疾病死亡人数展示为灰色条，仿佛笼罩战争的阴影。似乎战争本身没有那么可怕，是疾病让英国人措手不及，因而加剧了灾难。下一步是添加图例，用不同颜色区分各种死因，如图 12-2d 所示。（如果需要回顾 `legend()` 函数，请参考 3.2 节。）此外，图 12-2d 使用参数 `bty = "l"` 去掉了图周围的框。

表12-1：使用`plot()`或`lines()`绘制线的选项

参数	线的类型
`type = "p"`	点
`type = "l"`	线
`type = "b"`	线和开放点
`type = "c"`	点的位置上的有空隙的线
`type = "o"`	重叠绘制（即填充好点的线）
`type = "h"`	像直方图那样的垂直线

（续）

参数	线的类型
`type = "s"`	台阶
`type = "S"`	不同的台阶
`type = "n"`	不画图
`lty = "blank"`	
`lty = "dotted"`	
`lty = "dashed"`	
`lty = "dotdash"`	
`Lty = "longdash"`	
`lty = "solid"`	
`lty = "twodash"`	
`lwd = 1`	线宽。默认为 1。为较粗的线指定一个较大的值，或为较细的线指定一个较小的值

最后要谈到的这张图（见图12-3）绘制起来似乎有点麻烦，但放在这里有其用意。下面会详细解释这张图的创建过程，但如果你不想看，也可以直接跳到本节的最后一段。

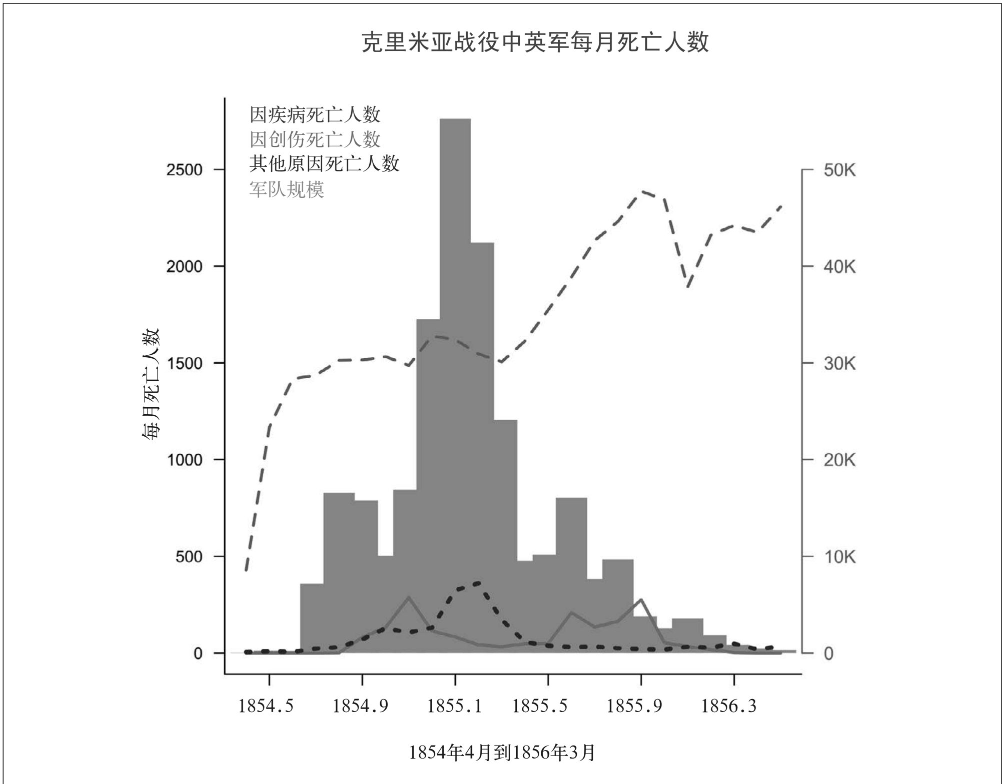


图 12-3：Nightingale 数据集的一个完整折线图（另见彩插）

以下改进使图 12-3 更为醒目和完善。

使用plot()函数中的参数main添加标题，同时为坐标轴添加标签

我们已经选择通过定义向量 t 然后在 plot() 中使用 main = t，把很长的 plot 命令变短。同理，这里也为标签创建了向量 x 和 y。

添加另一条线来表示每个月的军队规模

这有点棘手，因为军队的规模远远大于死亡人数。使用相同尺度会使 Army 数据超出图的范围，或使 Wounds 和 Other 太小，太接近水平轴，难以明显呈现出来。我决定将 Army 除以 20，并用右边的第二个垂直轴绘制新的结果变量，以显示部队的规模。绘制以不同尺度测量的变量，这可能令人疑惑或误解，所以这样做时要非常小心。在本例中，我特意让右边轴和左边轴的颜色、数字大小和线的类型完全不同，以尽量清楚地表明这是两个不同的尺度。

改进水平轴的呈现方式

对于变量 Date，图 12-4 只显示了两个 *x* 轴上的点。我为战争持续的 24 个月创建了新的变量 mon，值为 1~24。该变量在 *x* 轴上表示。数据集是按月排序的，因此这种方法可行。在 plot() 中，参数 xaxt = "n" 阻止打印 *x* 轴的标签，以便创建带标签的新坐标轴。它的标签将被指定为每年的三个月，足以提供足够的细节，但不会过多，以至于破坏可读性。参数 at 提供月份值，参数 labels 提供月份的名称。

增强的图 12-3 的创建脚本如下：

```
# 图12-3的脚本
# 若还没安装,则必须:
# install.packages("HistData", dep = T)
library(HistData)
attach(Nightingale)
par(mar = c(6,6,5,5), cex = .8) # 控制图片窗口的大小

Army2 = (Army)/20    # 缩小Army的尺寸使其适应图像
t = "British Army Deaths, Crimean War"   # 缩短plot命令长度
x = "Date, by Month, from April, 1854 to March, 1856"
y = "Number of Deaths per Month"
mon = 1:24  # 创建新的变量,比Date更易于处理

plot(mon, Disease,
  type = "h",
  lwd = 22,
  col = "gray67",
  lend = "butt",
  main = t,
  col.main = "maroon",
  ylab = y,
  xlab = x,
  cex.lab = .8,
  las = 1,
```

```
    cex.axis = .9,
    bty = "l",
    xaxt = "n")
  # xaxt = "n" 去掉x轴标签;使用axis()自定义坐标轴
  lines(mon,Wounds,
    pch = 18,
    col = "red",
    lty = "solid",
    lwd = 2)
  lines(mon, Other,
    lty = "dotted",
    col = "navyblue",
    lwd = 3)
  lines(mon, Army2,
    lty = "dashed",
    col = "seagreen4",
    lwd = 2)

  # 水平轴
  axis(1, at = c(2,6,10,14,18,22),
    labels = c("May 54","Sep 54","Jan 55","May 55","Sep 55", "Jan 56"))
  # 右边轴
  axis(4, at = c(0,500,1000,1500,2000,2500),
    labels = c("0", "10K","20K","30K","40K","50K"),
    las = 1,
    tick = T,
    cex.lab = .6,
    col = "seagreen4",
    col.axis = "seagreen4",
    ylab = "Troop Strength")

  legend("topleft", c("Death from Disease","Death from Wounds",
    "Other Deaths","Troop Strength"),
    text.col = c("gray40","red","navyblue","seagreen4"),
    bty = "n",
    cex = .8)

  detach(Nightingale)
```

本节阐述了如何创建复杂折线图。从前几个例子可以看出，构建此类图需要创建多层，并逐层应用，逐步制作一张复杂的展示图，使我们对最终的结果有更多控制。也许图 12-2b 已经满足了你的需求，不必烦劳绘制图 12-3。不过，你也可能想绘制出真正引人注目的视觉效果。有时，为了达到目的，你可能需要做所有令人厌恶的工作，但你最终可以绘制出真正漂亮的图。有些时候，你可以充分利用包的开发者已经完成的工作，正如我们在下一节中将会看到的那样。

12.3 模板

用 R 生成精细的基础散点图会有点麻烦。你也可以利用 `plot()` 函数中的很多可用的参数来定制图。你可以改变坐标轴，添加标题，改变图符，改变背景、标题和点的颜色，等等。如

前面提到的，这种定制有时工作量非常大。然而，一些包的设计者已经在他们的包中包含了模板或风格样式，使得各种风格实现起来很容易。我喜欢的一个风格来自 latticeExtra 包，它模仿了《经济学人》杂志的插图风格。以此风格重做图 12-1b，结果如图 12-4 所示。

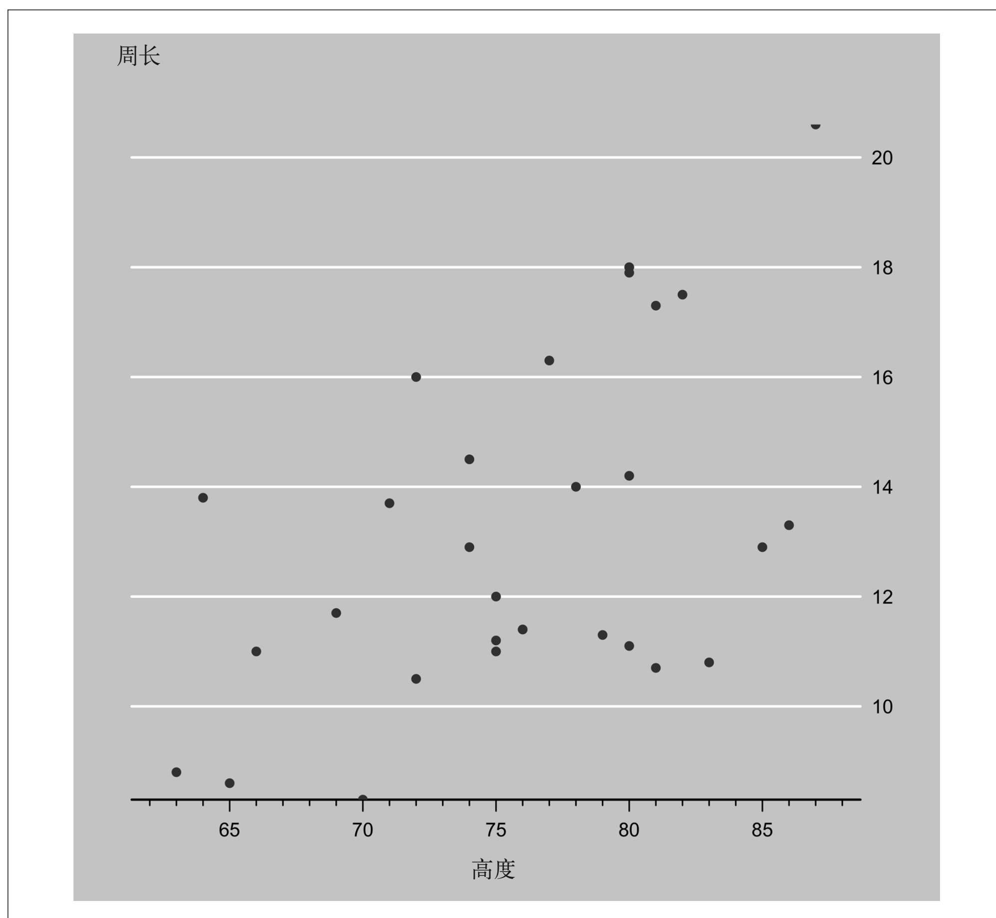


图 12-4：使用 asTheEconomist() 函数为 trees 数据制作的点阵图

这里用很少的代码获得了一张非常漂亮的图，代码如下：

```
# 使用模板来生成图12-4
# 若还未安装,则必须：
# install.packages("latticeExtra", dependencies = T)
library(latticeExtra)
attach(trees)
asTheEconomist(xyplot(Girth ~ Height), xlab = "Height",
  type = "p", with.bg = T)
detach(trees)
```

另一套模板是由 epade 包提供的。scatter.ade() 函数生成散点图。使用它的参数 wall，可

以从多个输出样式中进行选择。参数 `wall = 0` 生成的图与 `plot()` 函数生成的图相似，但这个参数值取 1~6 时则会创建其他有趣的图形。图 12-5 中有 6 张别致的图，每张只需要一行代码。还有关于点的颜色、文本、背景和线的参数，以及图例、线类型、点类型等的参数。欲获取更多信息，请输入 **`?scatter.ade`** 查询。

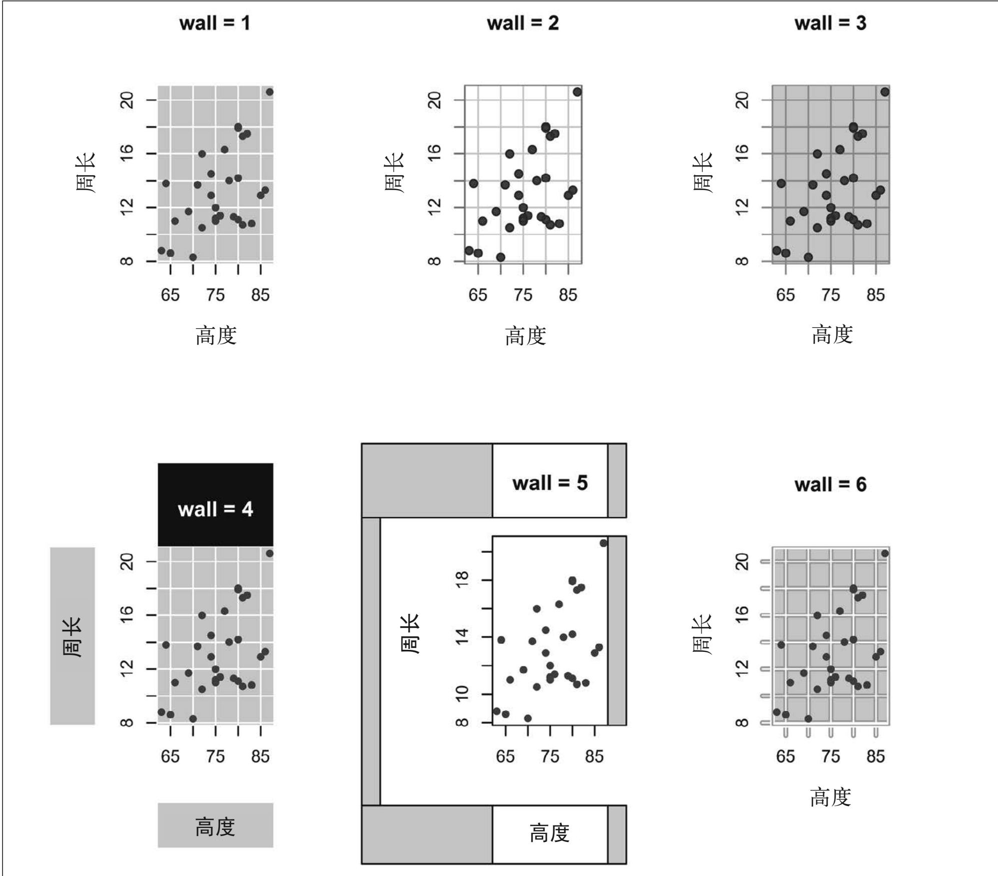


图 12-5：通过 epade 包中的 scatter.ade() 函数生成的散点图类型

下面的脚本显示了生成图 12-5 中的 6 张图是多么容易。如图 12-7 所示，它们的颜色完全不同，你可能喜欢也可能不喜欢这些风格：

```
# 图12-5
install.packages("epade") # 若还未安装
library(epade)
attach(trees)
par(mfrow = c(2,3))
scatter.ade(Height, Girth, wall = 1, main = "wall = 1")
scatter.ade(Height, Girth, wall = 2, main = "wall = 2")
scatter.ade(Height, Girth, wall = 3, main = "wall = 3")
```

```
scatter.ade(Height, Girth, wall = 4, main = "wall = 4")
scatter.ade(Height, Girth, wall = 5, main = "wall = 5")
scatter.ade(Height, Girth, wall = 6, main = "wall = 6")
detach(trees)
```

从头开始构建图 12-4 和图 12-5 中任意风格的图都要花费很多精力，需要编写大量代码。然而，包的开发者与我们分享了他们的劳动成果，我们可以使用他们提供的模板轻松生成图形。epade 包还为某些其他函数提供了相似的模板。在数以千计的 R 包中搜索可能会花费大量的时间，但这通常是值得的。

12.4　增强的散点图

要绘制散点图，除了 plot() 还有许多其他的 R 函数，其中一些提供了大量的增强功能。来自 car 包的 scatter plot() 就是一个很好的例子。图 12-6 展示了由该函数生成的图及以下附加信息：

- 空白处的箱线图，展示每个变量的分布；
- 回归线（直线）；
- 网格；
- 平滑的实线，以及散布测量值的虚线。

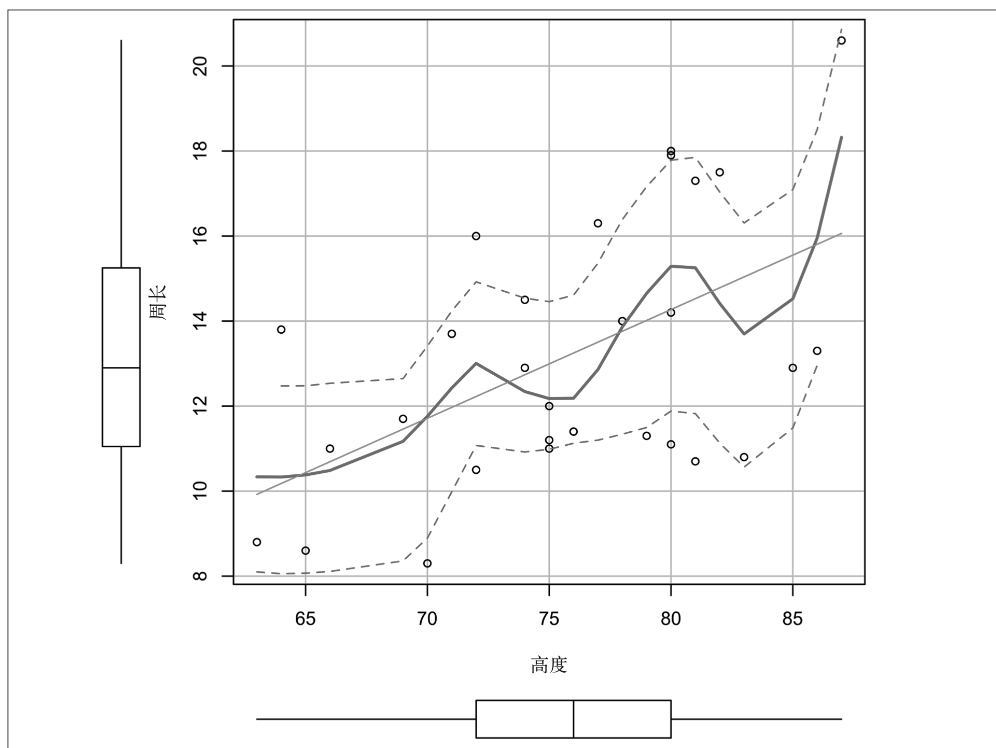


图 12-6：由 car 包中 scatterplot() 函数生成的周长和高度的散点图

平滑器（smoother）是一种工具，用于使散点图中数据的模式更易于识别。有几种类型的平滑器，但它们在给定 x 值（或相近的几个 x 值）处都显示了 y 的中心，连接这些点构成的线（通常是曲线）相对平滑。图 12-6 是一张使用了平滑器的散点图，平滑处理后的线为实线。你可以用参数 `smoother` 选择一种平滑方法，图 12-6 使用的是默认方法，即局部加权回归（loess 或 locally weighted regression）。

使用 `scatterplot()` 也可以轻松处理分组回归线、异常值的识别和其他工作，这些都超出了本书的范围。欲知更多信息，请输入 **`?sp`** 查询。生成图 12-6 的代码如下：

```
# 图12-6
library(car)
attach(trees)
sp(Height, Girth) # 注意缩写sp
detach(trees)
```

除了我们在上一节中看到的模板，`epade` 包中的 `scatter.ade()` 函数还有其他几个显著特点。它可以按组绘制数据，也可以绘制大小不同的点来代表幅度，还可以在图上放置线性回归、局部加权回归，或者多项式线，且处理图例相当容易。图 12-7 显示了用 `scatter.ade()` 为按照“处理”和“未处理”分组的数据绘图的一个例子。数据来自实验，想获取关于实验的信息，可以输入 **`?Puromycin`** 查询。

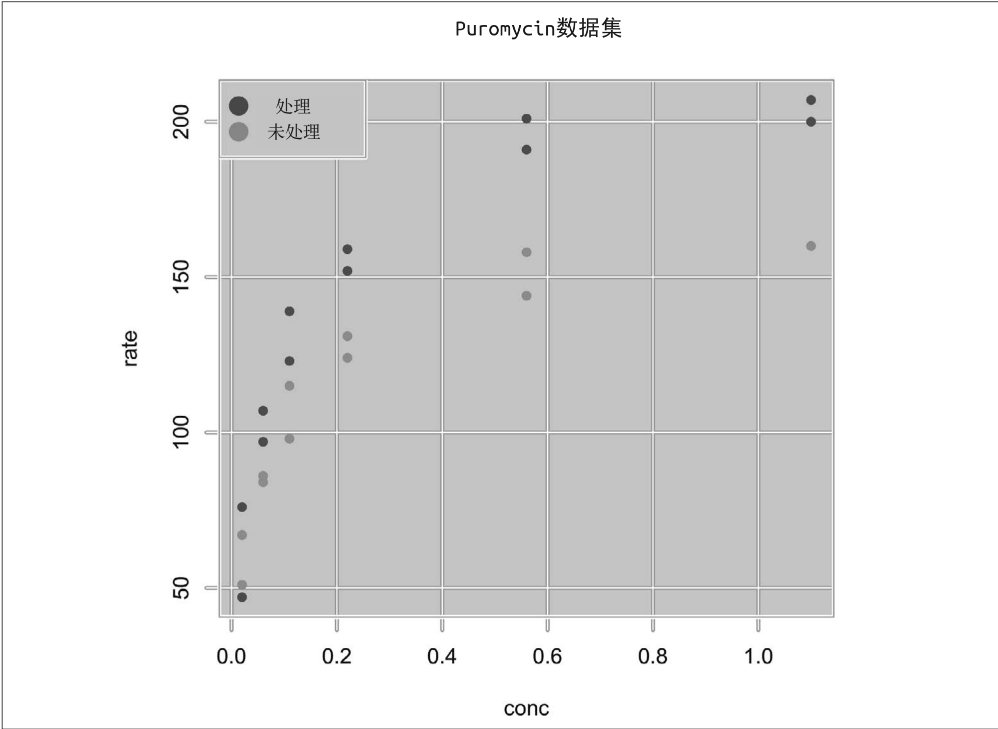


图 12-7：通过 `epade` 包中的 `scatter.ade()` 为 `Puromycin` 数据绘制的图

生成图 12-7 的代码如下：

```
# 图12-7
library(epade)
attach(Puromycin)
scatter.ade(conc, rate, group=state,
  col = c("royalblue3", "sienna1"),
  legendon = "topleft", wall = 6,
  main = "Puromycin dataset")
detach(Puromycin)
```

设计 `lattice` 包的目的是为了生成网格图，2.3.2 节曾提到过。这或许是回顾该节的大好时机。图 2-3 所示为网格散点图的例子。图 12-8 是用 `lattice` 为 `Puromycin` 数据绘制的图，在不同的窗口或面板（panel）分别展示处理和未经处理的对象。请将其与图 12-7 进行比较。

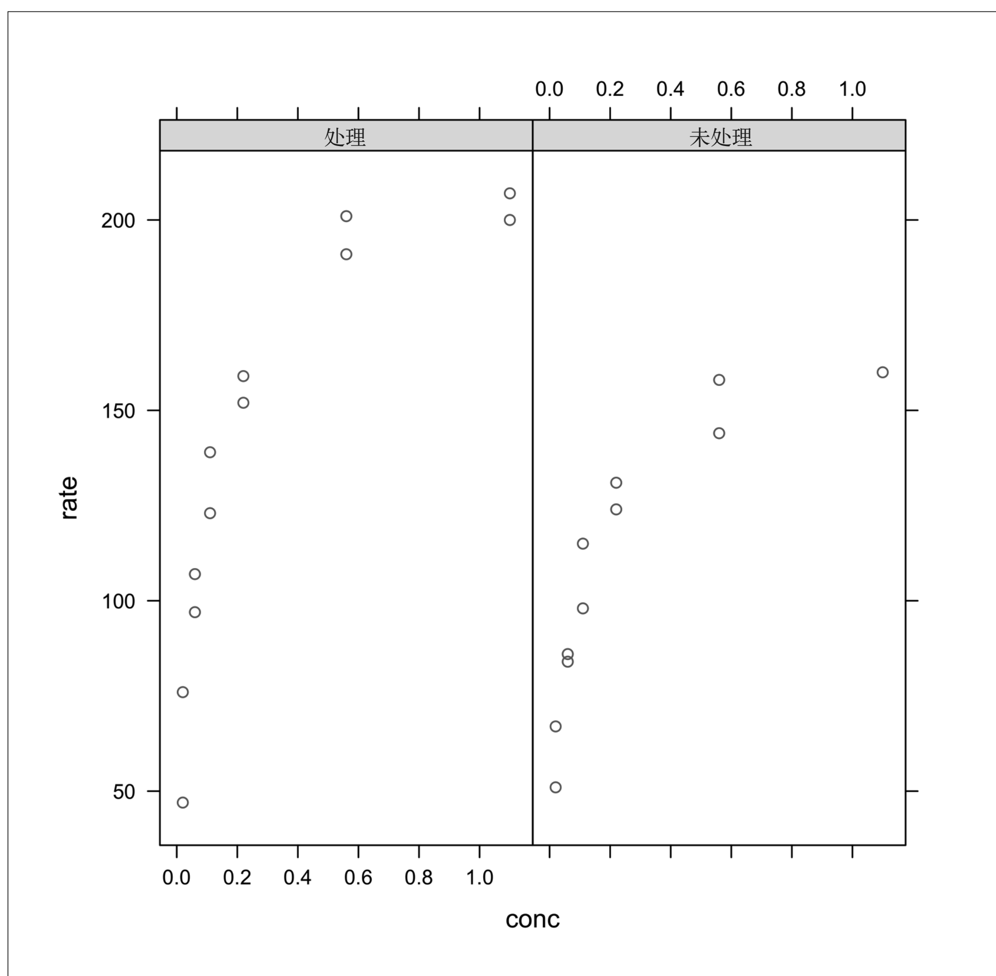


图 12-8：`Puromycin` 数据通过 `lattice` 包中的 `xyplot()` 绘制成网格图

生成图 12-8 的代码如下：

```
# 图12-8
library(lattice)
attach(Puromycin)
xyplot(rate ~ conc | state)
detach(Puromycin)
```

散点图或许是最有用和最常用的图。其他许多种类的图（包括后续章节中的几种图）都是基于散点图的。R 为这类图提供了很多实现，其中一些已经在本章中讨论过。细想一下，本章各式各样的散点图都是由 R 绘制的，也许本章的图函数对你来说就够用了，但如果你愿意搜索，在 R 中还能找到更多的散点图函数。

x 和 y：为什么

有些生成散点图的函数，比如 plot() 和 scatter.ade()，希望 x 和 y 的变量名以 x,y 格式生成列表。另一些函数，尤其是 xyplot()，则期望得到 y~x 或 y~x | z 形式的**公式**，其中 z 是条件变量。还有一些函数，如 scatter plot()，可以接受以上任意一种变量名。其实，plot() 也是如此，尽管它的帮助文件没这么说。许多（或许是大多数）R 函数都能接受其中一种形式或两种都能接受，有时也包括相近的变体。

12.4.1　练习12-1

如果保存了练习 1-2 中使用的 emissions 数据集，现在你可以使用如下命令检索它：

```
> load("emiss.rda")
```

如果没有保存，现在输入部分数据，建立三个向量：

```
> Year = c(2004:2010)
> Europe = c(7.9, 7.9, 7.9, 7.8, 7.7, 7.1, 7.2)
> Eurasia = c(8.5, 8.5, 8.7, 8.6, 8.9, 8, 8.4)
```

请绘制一张折线图，展示欧洲和欧亚大陆七年间的排放量。绘制不同的线，通过颜色和线类型进行区分，并包含图例。你的图应该和图 12-9 类似。

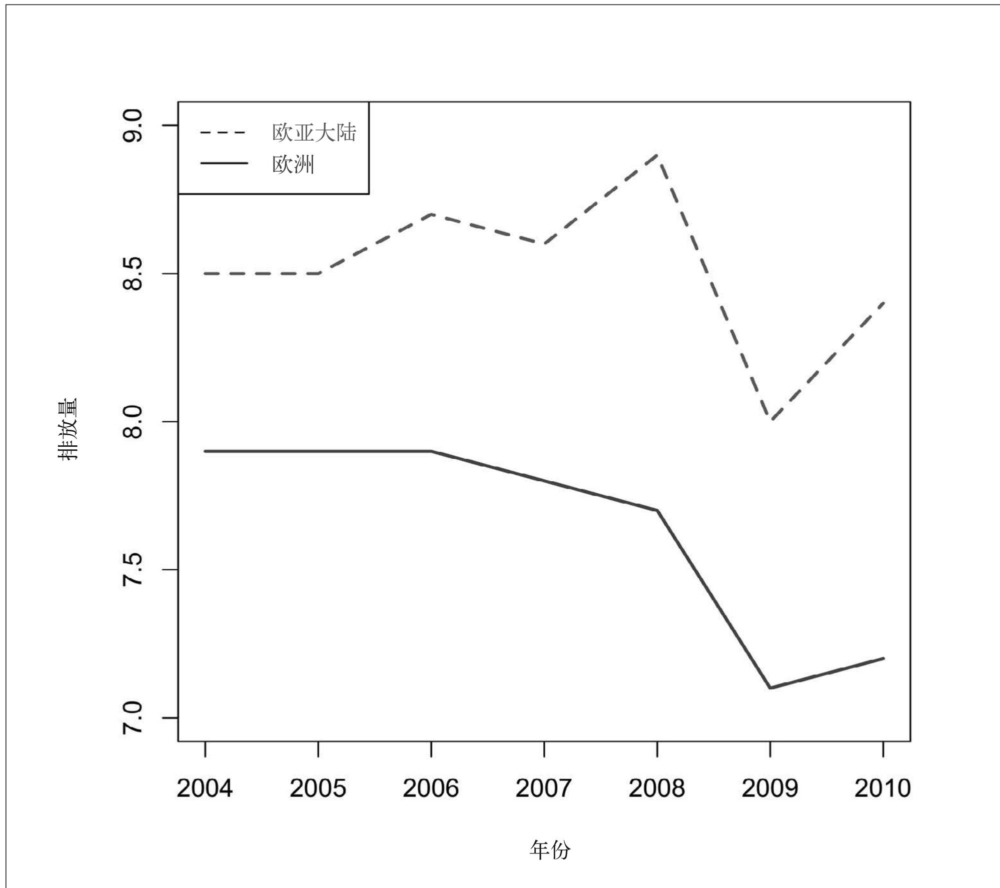


图 12-9：欧亚大陆和欧洲排放量的折线图

12.4.2 练习12-2

为银河系之外星云的 `Velocity`（速度，x 轴）和 `Distance`（距离，y 轴）简单绘制一张图，你可以在来自 `sleuth2` 包的 `case0701` 数据集中找到它。这两个变量之间有什么关系？使用你自己选择的模板来制作更有趣的数据展示。

第 13 章

高密度图

处理大数据集

有时，在大数据集上应用散点图等技术可能很难。以来自 car 包的一个数据集为例，Vocab 中有 21 000 多个观测值，包括一些基本的人口统计数据和词汇测试成绩。加载包并查看数据（使用 head() 命令时要当心，一不小心就会打印整个数据集），代码如下：

```
> library(car)
> attach(Vocab)
> head(Vocab)

         year    sex education vocabulary
20040001 2004 Female         9          3
20040002 2004 Female        14          6
20040003 2004   Male        14          9
20040005 2004 Female        17          8
20040008 2004   Male        14          1
20040010 2004   Male        14          7
```

vocabulary（词汇量）与 education（教育水平）的关系值得探讨。低学历的人词汇成绩偏低，分数随着受教育水平的提升而提升，这种看似合理的预期是否符合实际情况？散点图应该可以解答这个问题，创建图的方法如下：

```
# 图13-1
library(car)
attach(Vocab)
plot(education, vocabulary)
detach(Vocab)
```

图 13-1 中的散点图并没有呈现清晰的趋势！没有线条或点形成的带鲜明地体现出我们期望看到的关系。该图左上角和右下角有少量空白，但其他地方的粒子数看起来相同。

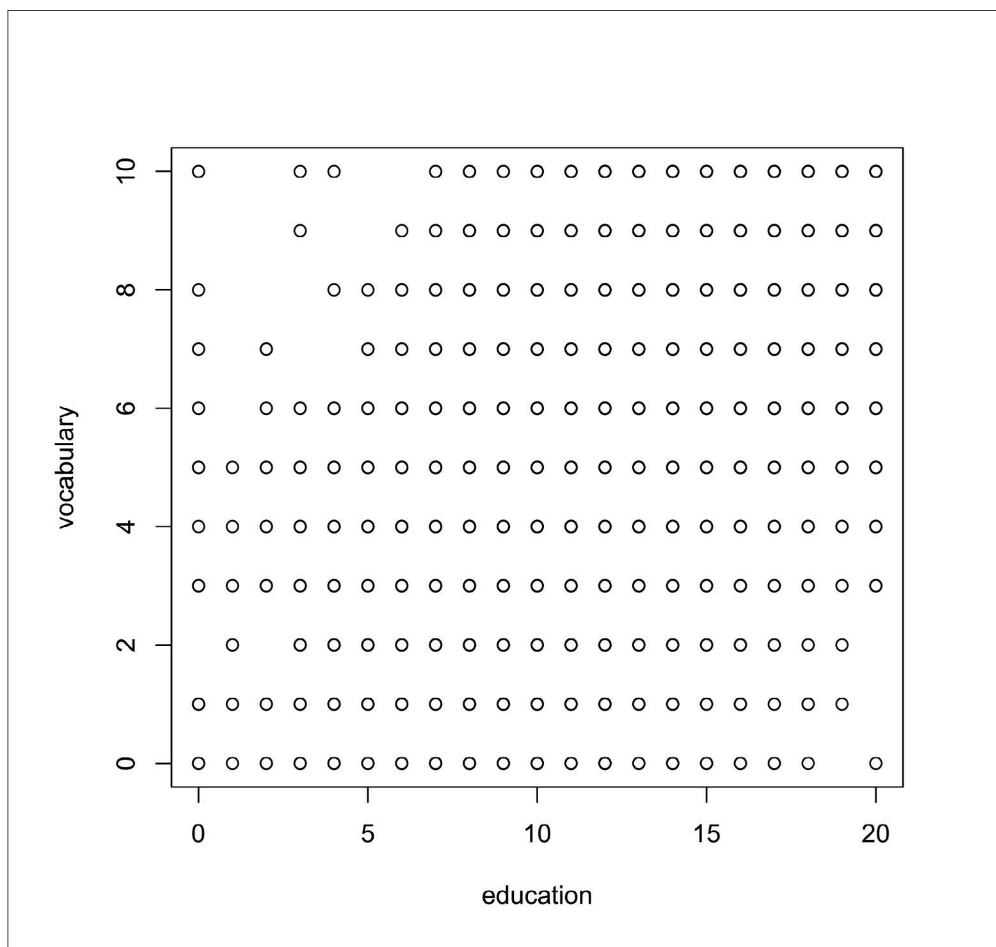


图 13-1：education 和 vocabulary 的散点图

这两个变量是离散的（discrete），也就是说，尽管它们是数值型而不是类别型，但在其数值范围内仍然只取有限的几个值。教育水平是以年为单位的整数，一个人可以接受 12 年的教育，而不能接受 12.4 年或 10.75 年的教育。词汇量则用回答正确的题目数量来测量。vocabulary 仅有 0~10 这 11 个值，而 education 只取 0~20 这 21 个值，图上仅有 11 × 21 = 231 个位置可以出现点，但参与调查的人数为 21 000 多个。当然，这意味着图上有多处叠加。我们能做些什么呢？

在第 3 章中，我们用称作“抖动”（jittering）的技巧处理带状图中类似的问题。这或许也可以在此发挥作用。然而，如果点上下跳动，那么词汇成绩看起来就不像整数了，这样一

来，我们可能需要进行更精确的测试。让我们用带有参数 jitter 的 scatterplot() 尝试一下，代码如下：

```
# 图13-2
library(car)
attach(Vocab)
sp(education, vocabulary, jitter = list(x = 2, y = 2),
   smoother = F, spread = F, reg.line = F)
```

结果如图 13-2 所示。

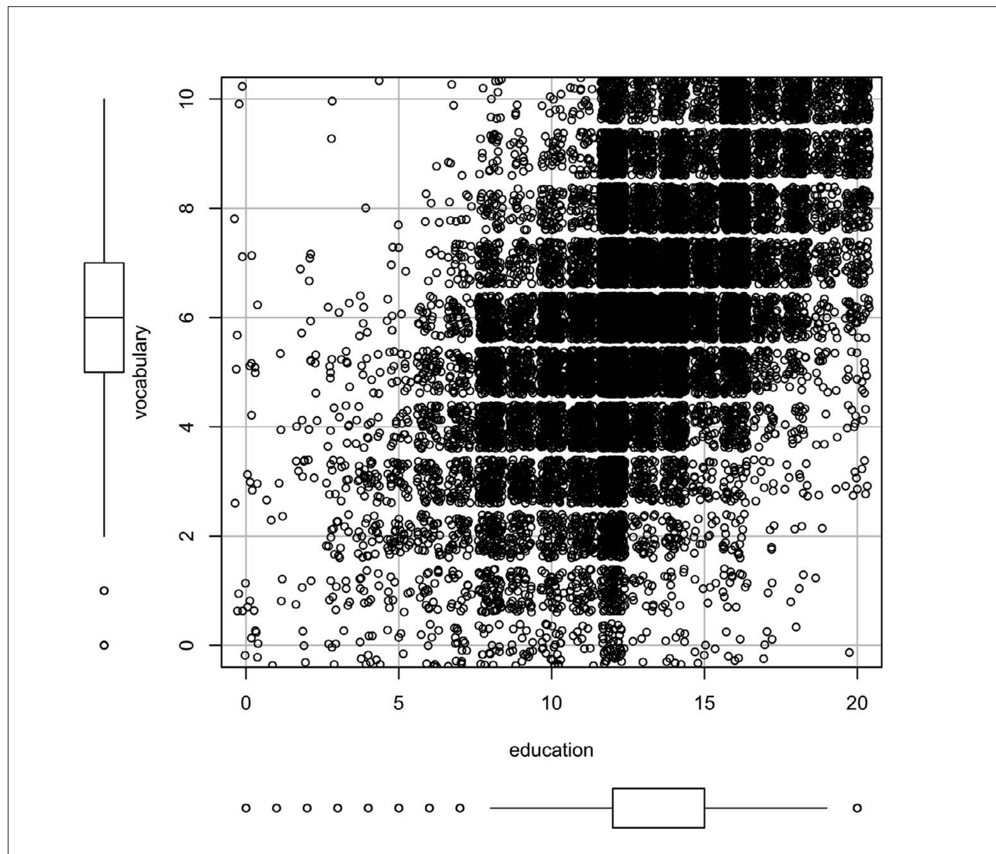


图 13-2：带有抖动的散点图

图 13-2 对图 13-1 做了不少改进，现在我们可以看到一个清晰的图案。使用下面的参数可以控制抖动量：

```
jitter = list(x = 2, y = 2)
```

试着把抖动量从 2 改为其他值，看看会有什么样的效果。

前文中提到过另一种方法，就是采用较小的绘图符号，但对于本例，这个技巧并不合适。它不会将这张图中的点分开，只能使它们更小。然而，缩小点并同时增加抖动，可使其更清楚一点。你可以尝试一下。

向日葵图

你还可以绘制向日葵图（sunflower plot）。根据点的数量，这种类型的图在不同位置使用不同的符号。来看下面的代码：

```
# 图13-3
library(car)
attach(Vocab)
sunflowerplot(education, vocabulary,
   main = "Sunflower Plot",
   col.main = "deepskyblue3",
   family = "HersheySerif",
   font.lab = 3) # x 和 y标注为斜体
detach(Vocab)
```

结果如图 13-3 所示。

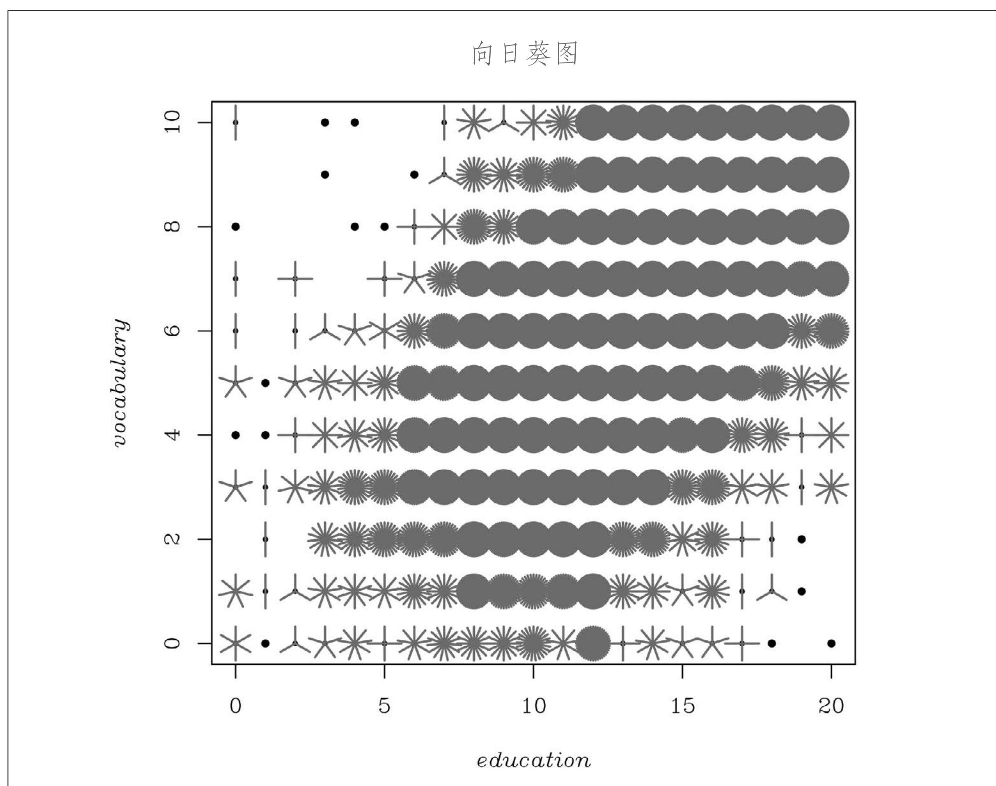


图 13-3：education 和 vocabulary 的向日葵图

图 13-3 中的向日葵图和抖动的散点图看起来有点相似。思考一下向日葵图怎样展现数据。看看左上角的点，那是一个带有两片花瓣（petal）的黑点，代表了两个观测值。（实际上在参加调查的 21 000 人中，有两个人没有接受教育，却在词汇测试中拿到了最高分！）而左下角的点说明有一个受过 20 年教育的人（博士？硕士？双硕士学位？）在词汇测试中没有答对任何问题。这是否体现了数据的质量？是否出现了错误？回到图的左上方，从顶部向右边移动，我们会看到接下来的两点，它们各代表一个观测值，接着有一个点带两个花瓣，有一个点带九个花瓣（代表九个观测值），再下一个点有三个花瓣，等等。灰色的实心圆代表许多的观测值，多到我们可以不再计算花瓣数量。

虽然这张图仍然没有提供理想的视觉分辨率，但它确实相当好地表明了图中任意位置上点的密度。看来预期的受教育程度和词汇量之间的关系是正确的。

改变字体

你是否注意到图 13-3 中的字体发生了改变？R 中有许多可供选择的字体。你可以使用 `par()` 命令中的参数 `family` 请求字体，它会影响之后给出的任何命令，直到发出一个新的 `par()` 命令。对于用一条命令创建的图，大多数绘图命令也会接受参数 `family`。关于可用字体的更多信息，请输入 **`?Hershey`** 或者 **`demo(Hershey)`** 查询。

另外，你可以通过参数 `font`（或者 `font.axis`、`font.lab`、`font.main`、`font.sub`）指定字体类型：

- 1 = 纯文本（plain text）
- 2 = 粗体（bold）
- 3 = 斜体（italic）
- 4 = 粗斜体（bold italic）
- 5 = 符号字体（symbol font）

注意，并不是所有的字体家族都包括所有的字体类型（比如粗斜体、超粗体、压缩体）。欲知更多信息，请输入 **`?Hershey`** 进行查询。

平滑散点图

在 R 中，有更好的图形化工具可以处理高密度数据的问题。`smoothScatter()` 函数就采用了不同的方法，代码如下：

```
# 图13-4
library(car)
attach(Vocab)
smoothScatter(education, vocabulary,
  las = 1,
  family = "HersheyGothicGerman",
  main = "Smooth Scatter Plot", font=3)
# las = 1 旋转y坐标轴上的数字
detach(Vocab)
```

结果如图 13-4 所示。

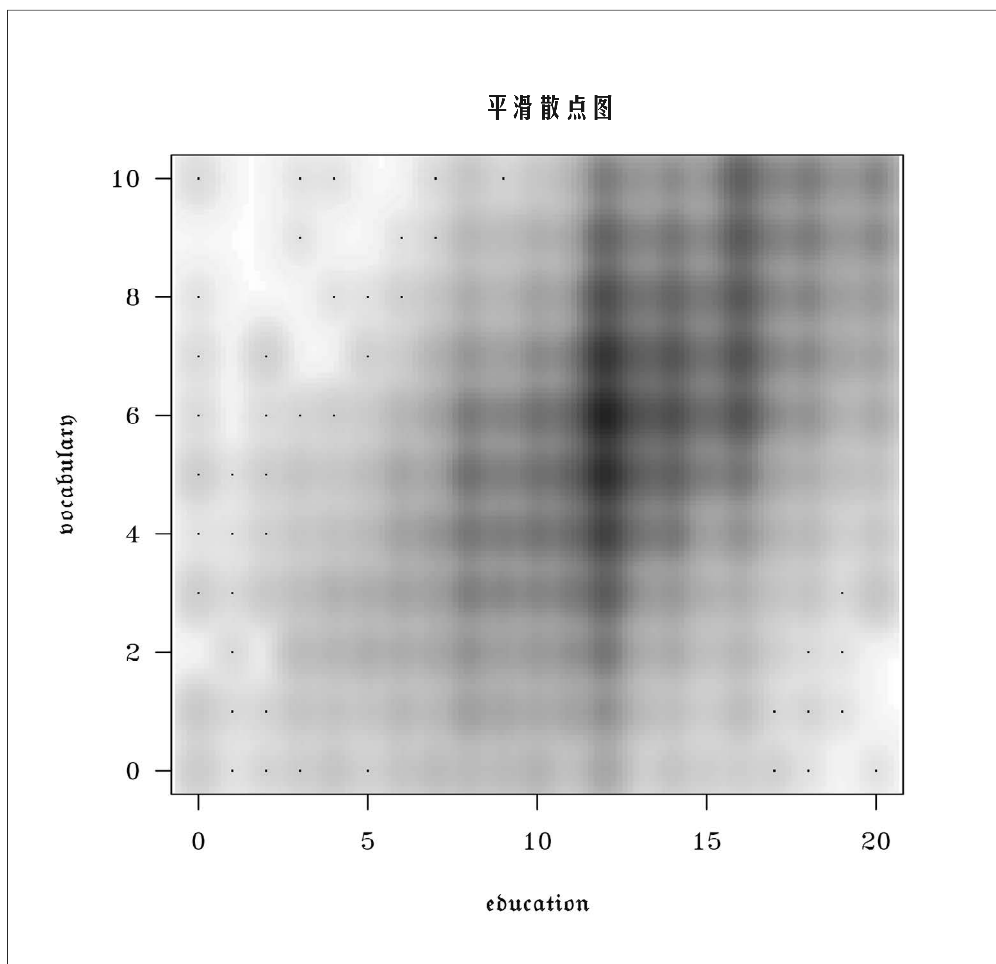


图 13-4：`education` 和 `vocabulary` 的平滑散点图

图 13-4 的平滑散点图使用色调和颜色强度区分高密度区与低密度区。这不仅比向日葵图更加美观，而且也提供了更好的分辨率。例如，图中最大的黑点表示，受过大约 12 年教育（高中毕业）的人大多取得了 5~7 分的词汇成绩。受过大约 14 年教育（社区学院）和 16 年教育（大学毕业生）的人在图中构成了黑色的带状区域。在向日葵图上是看不到这些情况的。

向日葵图很好地展示了主要趋势，甚至比平滑散点图更胜一筹。而平滑散点图显示了向日葵图可能错过的细节。在研究数据的时候，以几种不同的方式来整理数据通常是一个好主意。尽管数据最后可能以平滑散点图来呈现为佳，但也不一定总是如此。

hexbin图

大数据集还可以选择进行分组（binning）。这和向日葵图相似，但重叠的点表现为区间（bin）中的总数，而不是点的形状。`hexbin` 包中的 `hexbin()` 函数可以实现分组，图 13-5 展示了一个例子。这里可以使用多种颜色梯度，比如 `colramp =` 选项。（获取更多信息，请输入 **`?colorramps`** 查询。）示例中使用了颜色梯度 BTC。请注意，右边说明了特定颜色表示的数字。还有许多选项可用，比如平滑处理、网格 hexbin、添加直线、hexbin 图矩阵等。获取更多信息，请输入 **`?hexbin`** 查询。

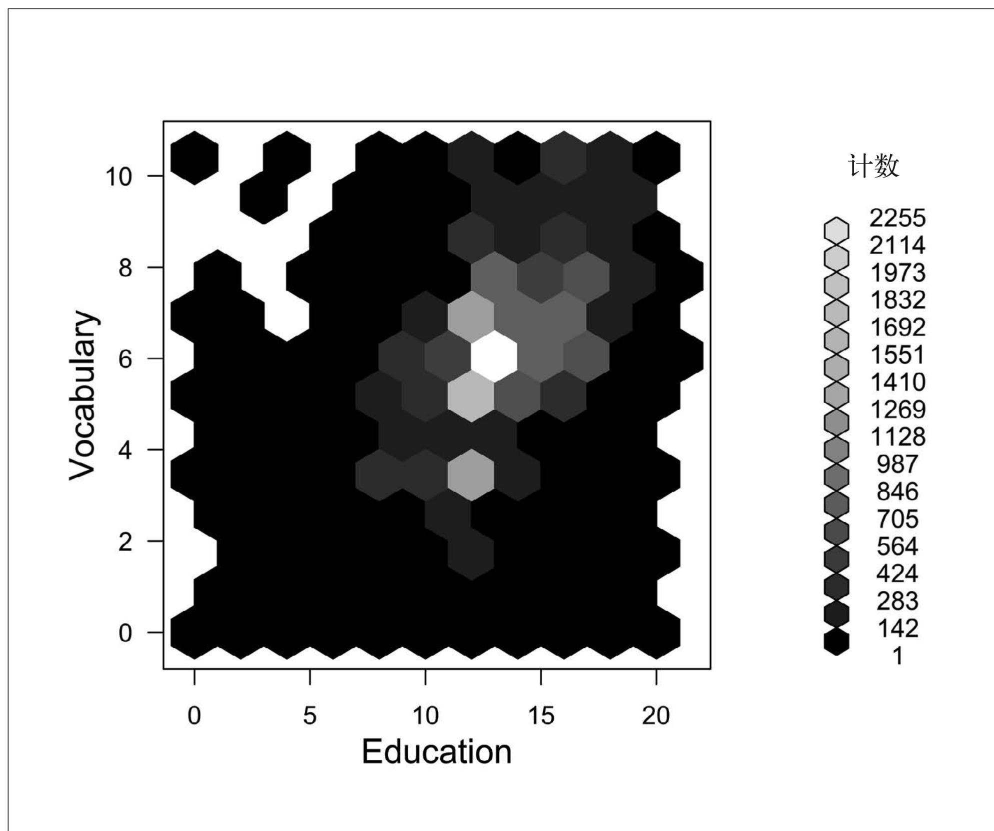


图 13-5：使用 `hexbin()` 函数分组的例子（另见彩插）

生成图 13-5 的代码如下。

```
# 图13-5,hexbinning
install.packages("hexbin", dependencies = T)
library(car)
library(hexbin)
plot(hexbin(Vocab$education, Vocab$vocabulary, xbins = 10),
  xlab = "Education", ylab = "Vocabulary", colramp = BTC)
```

参数 `xbins` 并不总是必要的，尽管在本例中，它非常有帮助。尝试不使用参数 `xbins` 绘制该图，看看发生了什么。`xbins` 的默认值（沿 *x* 轴的区间数）为 30。参数 `xbins = 10` 使区间宽度足以填补空白。数字越大，区间越小，离得越远；反之，数字越小，区间越大，离得越近。

如果没有受到下述限制，平滑散点图也许更加令人满意。首先，数据是离散的，只包括整数，所以只能有 11 个词汇等级和 21 个教育水平。其次，为了让相近的颜色易于辨别，色彩渐变的设计只有这么多的层次。因此，这里最多只能有 16 级颜色。

hexbin 图和平滑散点图是伪色图（false-color plot）的两种类型，这类图使用颜色梯度表示数量和强度。本书后面要讨论的热图（heat map plot）是另一种类型的伪色图。

练习13-1

为 `nlme` 包中的 `MathAchieve` 数据集的 `MathAch`（*y* 轴）和 `SES`（*x* 轴），绘制一个简单的散点图。图中是否存在明显的趋势？其他类型的图是否可以更好地揭示数据间的关系？尝试绘制本章中介绍的这几种图。哪种最有启发，哪种最模糊不清？

第14章

Bland-Altman图



如果对编程不感兴趣，你可以跳过本章，不必担心错过什么。

评估测量的可靠性

本章介绍R的灵活性，在基础R中不可能找到此处研讨的图表类型，在成千上万的可用包中找到这样一个专门的类型也要花费精力。我决定编写自己的函数来完成这项任务，并已在书中介绍。编写函数的过程中，我发现至少有两个包有该图，本章会介绍其中的一个。本章涉及的输入比其他大多数章节稍多，需要将一个数据集输入电子表格或文本文件，还要输入一个相对较长的R函数。本章结尾介绍了一个较短版本的函数，如果你愿意，可以替换使用。

Bland-Altman图用于评估两项测量技术之间的一致性，或者测量的可靠性/可重复型。它也被称为“Tukey平均差图”。

Bland和Altman（1986）给出了表14-1中列出的最大呼气流速（PEFR）的测量数据，数据以升/分为单位，测量使用了两种不同类型的计量器——Wright流计量器和迷你Wright流计量器。每个流计量器对同一对象进行两次测量，目的是确定迷你流计量器和Wright流计量器的读数是否基本相同，从而用前者取代后者进行标准测量。注意，这不是一个相关问题（详见16.2节）。在这里，两种方法密切关联，甚至完全相关都是不够的；测量本身必须是可互换的。也就是说，我们需要知道两个流计量器是否能够得到相同的结果。

表14-1：Bland-Altman PEFR数据

对象	Wright1	Wright2	迷你1	迷你2
1	494	490	512	525
2	395	397	430	415
3	516	512	520	508
4	434	401	428	444
5	476	470	500	500
6	557	611	600	625
7	413	415	364	460
8	442	431	380	390
9	650	638	658	642
10	433	429	445	432
11	417	420	432	420
12	656	633	626	605
13	267	275	260	227
14	478	492	477	467
15	178	165	259	268
16	423	372	350	370
17	427	421	451	443

将数据输入电子表格，并保存为以 BlandAltmanPeakFlow.csv 为名的 .csv 文件。使用如下命令，把该数据读取到名为 `Flow` 的 R 数据框：

```
Flow <- read.csv("BlandAltmanPeakFlow.csv", header=TRUE)
```

接着使用下面的命令，核实数据和表 14-1 中的数据是否相同：

```
head(Flow)
```

下列代码中的 R 函数将生成图 14-1 中展示的 Bland-Altman 图。这就是一张 17 个点的散点图，一个点对应研究中的一个对象。图上的点 (x,y) 被定义为 x = 一个对象的两个测量值的平均值，y = 两个测量值之间的差值。函数定义如下：

```
# 图14-1中的结果,本章末有较短的版本
baplot <- function(meas1, meas2){

# 计算平均值和偏差
ave = (meas1 + meas2)/2
   dif = meas1 - meas2

# 为参考线计算参数
   std = sd(dif)
   mdif = mean(dif)
   mdrnd = round(mdif,3)
   mxav = max(ave) - (max(ave) - min(ave))/12
```

```
    upperci = round((mdif + std*1.96), 3)
    lowerci = round((mdif - std*1.96), 3)
    maxx = 1.05*(max(ave))
    minx = 1.05*(min(ave))
    maxy = max(upperci,max(dif))
    miny = (min(lowerci,min(dif)))

# 绘制点
plot(ave,dif,
  pch = 16, col = "deepskyblue3",
  xlim = c(minx,maxx), ylim = c(1.1*miny, 1.1*maxy),
  main = "Bland-Altman Plot",
  col.main ="deepskyblue4",
  xlab ="Average of two methods",
  ylab ="Difference between two methods", las=1)

# 绘制参考线
abline(h = mdif,
  lty = "solid", col = "grey75", lwd = 2)
abline(h = mdif + std*1.96,
  lty ="dotted", col = "grey75", lwd = 2)
abline(h = mdif - std*1.96,
  lty = "dotted", col = "grey75", lwd = 2)

# 在参考线周围添加文本
text(mxav, mdif,
  labels = "mean difference",
  pos = 3,
  cex = .7)
text(mxav, mdif,
  labels = mdrnd,
  pos = 1,
  cex = .7)
text(mxav, upperci,
  labels = "upper 95% limit of agreement",
  pos = 3,
  cex = .7)
text(mxav, upperci,
  labels = upperci,
  pos = 1,
  cex = .7)
text(mxav, lowerci,
  labels = "lower 95% limit of agreement",
  pos = 3,
  cex = .7)
text(mxav, lowerci,
  labels = lowerci,
  pos = 1,
  cex = .7)
}
```

把函数保存到文件的代码如下：

```
save("baplot",file = "baplot")
```

在输入并保存了数据和函数之后，你可以开启 R 会话并绘制该数据的图（如图 14-1 所示），比较 wright1 和 mini1。使用如下命令来实现该操作：

```
# 图14-1
Flow <- read.csv("BlandAltmanPeakFlow.csv", header = TRUE)
load("baplot")
baplot(Flow$wright1, Flow$mini1)
```

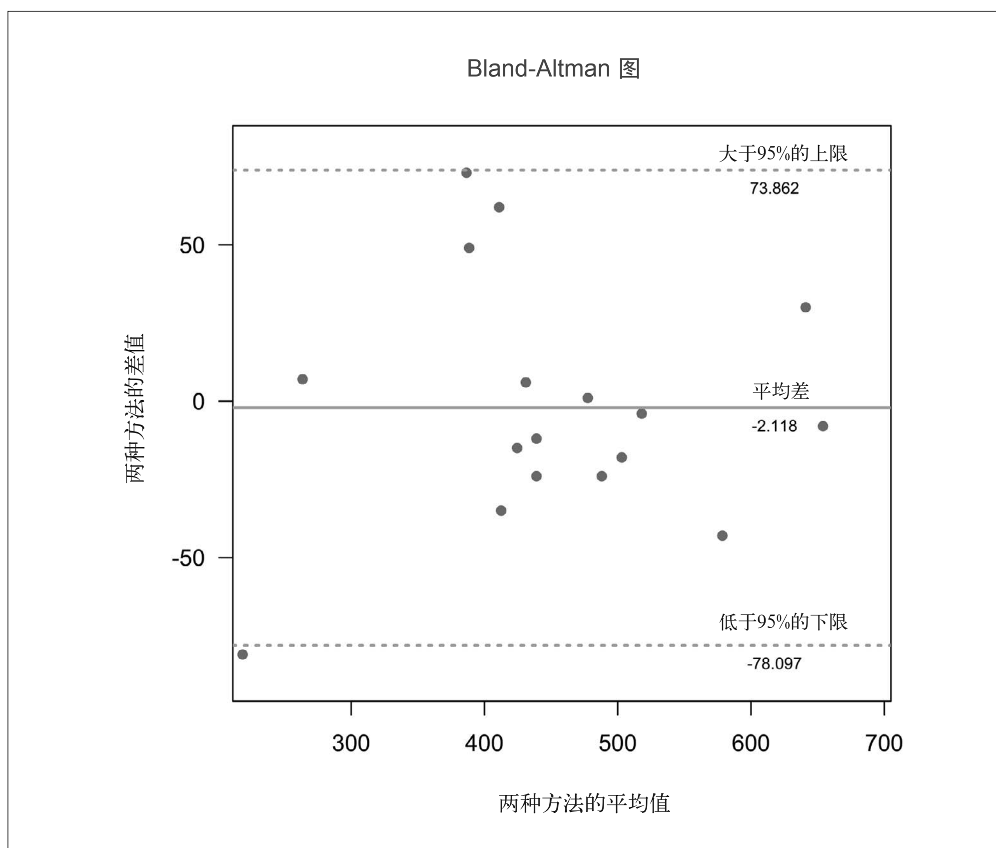


图 14-1：PEFR 数据的 Bland-Altman 图，使用 baplot() 函数生成

图 14-1 为每一个对象显示了一个点。点代表了对象的两个测量值的平均值（水平轴）以及两个测量值间的差值（垂直轴）。图中有三条参考线。实线表示平均差（在比较研究中称为偏差）。虚线表示协议限制（limits of agreement）。要计算协议限制，首先要找出差异的平均值（称之为 m）和差异的标准差（称之为 s）。因此，上限和下限如下：

```
m ± 1.96 * s
```

如果你更喜欢有参考线而没有标签的图，在 baplot() 函数中删除 text 表达式即可。



图 14-1 中的值和 Bland 和 Altman（1986）中值的差异是舍入误差造成的。`baplot()` 函数小数位更多，而且使用标准差为 1.96 的精确乘数，而不是四舍五入为 2。在实践中，这些差异是微不足道的。

如果没有系统的偏差，点会围绕平均值参考线聚集成带。协议限制应不大于临床可接受的误差。这个例子中的协议限制相当大，最大接近 80 升 / 分，临床上不可接受。两种方法之间的差异可能与两个测量的平均值有关。注意，平均值为 400 附近的集群的差异较大，平均值约为 200 处有一极值。看来迷你流计量器不能进行替代标准方法。然而，这个结论基于一个非常小的样本。这里的明显差异在更大的样本中可能不会那么明显。

这种图的另一种用途是比较针对同一对象的重复测量。例如，你可以通过为变量 `wright1` 和 `wright2` 绘图来研究标准测量方法的可靠性。这一点至关重要。如果 Wright 流计量器对同一对象的测量互不相同，那么试图评估与迷你流计量器的一致性是没有什么意义的。生成这种图的代码如下：

```
# 图14-2
baplot(Flow$wright1, Flow$wright2)
```

重复测量的实验结果如图 14-2 所示。

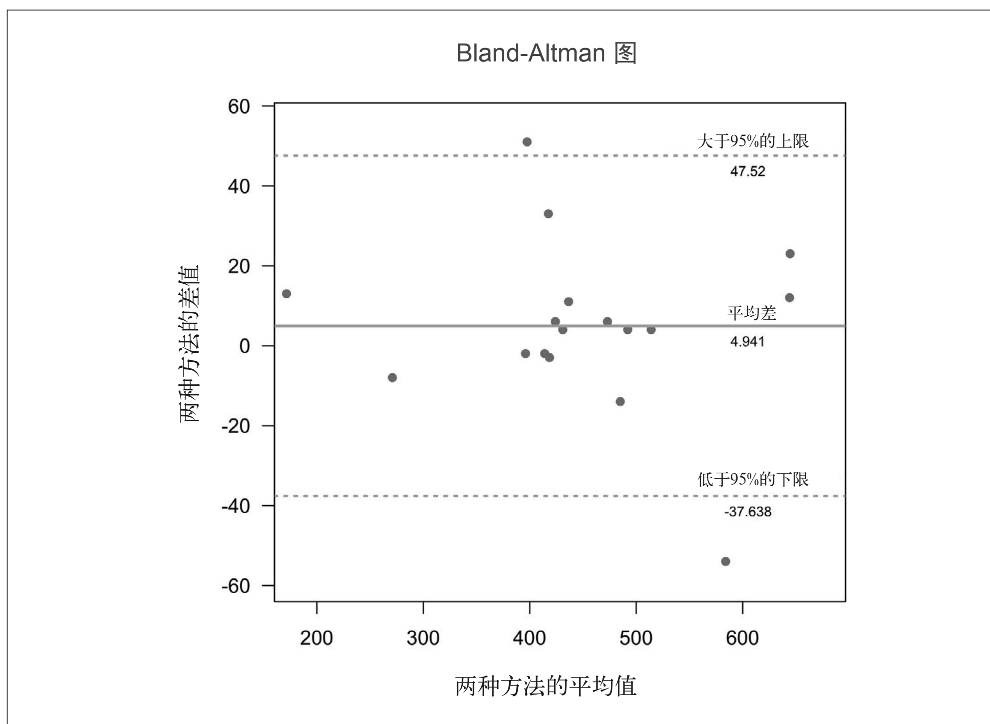


图 14-2：`wright1` 和 `wright2` 的对比。每一个对象都用同一种仪器测量两次

大多数差异是相当小的，17 个对象中有两三个显示出约 ±50 的差异。该结果暴露了流测量的可靠性问题。对更大的样本进行研究也许能进一步解答这个问题。

epade 包提供了几个绘图工具，其中一个工具生成的 Bland-Altman 图和我们的图非常相似，实现代码如下：

```
# 图14-3
install.packages("epade") # 若还未安装
library(epade)
Flow <- read.csv("BlandAltmanPeakFlow.csv", header = TRUE)
bland.altman.ade(Flow$wright1, Flow$mini1, fitline = 0)
```

比较一下图 14-3 和图 14-1 的结果。该函数也允许在图上放置回归线，可以是线性的（linear）、多项式的（polynomial）或者局部加权回归的（loess）。要查看帮助文件，请输入如下命令：

```
> library(help = epade)
```

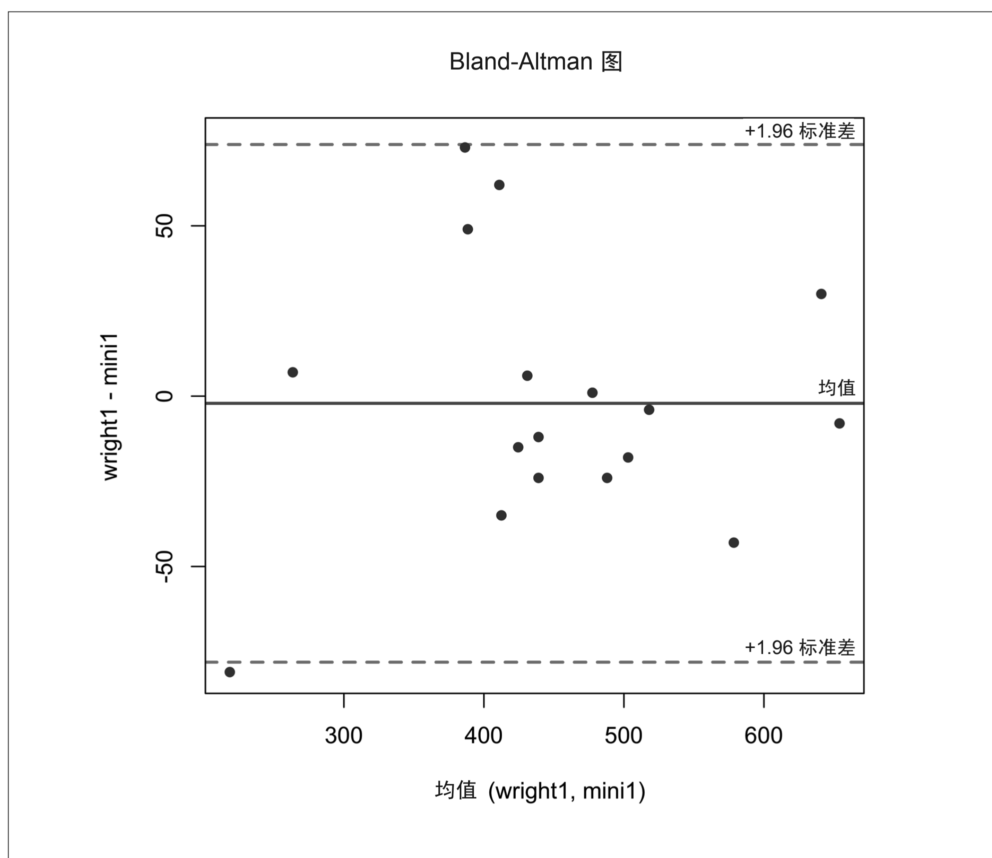


图 14-3：来自 epade 包的 Bland-Altman 图

在本章中，我们已经看到了如何绘制基础 R 包中不提供的图类型。你可以编写并保存一个函数，多次使用它实现绘图。我们也比较了一个包中提供的类似但并不完全相同的图，你可以下载这个包并将其添加到 R 中。这表明了 R 的灵活性和可扩展性。由此可知，为工作选择合适的工具时要谨慎。`baplot()` 的例子简单演示了如何编写一个函数。需要注意的是，`baplot()` 也有局限性，例如，它不能在缺少值的情况下正常运行，而且也不提供任何选项（比如在图上添加回归线）。如果你要求不多，用这个函数就很好。你可以将 `baplot()` 和一个更复杂的函数（比如 `epade` 中的函数）进行对比，这会很有趣。输入没有括号的函数名称，可以查看其代码（当然，前提是你已经安装并加载了 `epade`），如下所示。

```
bland.altman.ade   # 展示此函数的代码
```

练习14-1

选择本章介绍的一张 Bland-Altman 图，为 `ResearchMethods` 包中的 `MFSV` 数据绘图。你的结论是什么？两种方法可以互换吗？

较短版本的baplot()

如下代码是较短版本的 `baplot()`，没有那两条参考线。你或许想以更少的输入尝试用户自定义的函数。

```
baplot <- function(meas1,meas2){
ave = (meas1 + meas2)/2
dif = meas1 - meas2
mdif = mean(dif)
plot(ave,dif,
  pch = 16,
  main = "Bland-Altman Plot",
  xlab = "Average of two methods",
  ylab = "Difference between two methods", las = 1)
abline(h = mdif,
  lty = "solid", col = "grey75", lwd = 2)
}
```

第15章 QQ图

比较数字集合

比较两个数字集合的分布可能是非常有帮助的，例如，比较两个变量或两个向量。两个数字集合可能都是测量值集合，或者其中一个可能是理论分布。例如，我们可能希望知道特定变量与理论上的“正态分布”的比较情况会是怎样。

在美国和世界上其他的许多地区，顾客给服务人员小费是一种习俗。在餐馆的食客之间，给多少小费是一个经常讨论的话题。reshape2 包中有一个 tips 数据集，是一名服务员整理的自己的小费清单。看看这个有趣的数据集，如下所示：

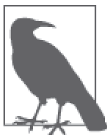
```
> library(reshape2)
> attach(tips)
> head(tips)
  total_bill  tip    sex smoker day   time size
1      16.99 1.01 Female     No Sun Dinner    2
2      10.34 1.66   Male     No Sun Dinner    3
3      21.01 3.50   Male     No Sun Dinner    3
4      23.68 3.31   Male     No Sun Dinner    2
5      24.59 3.61 Female     No Sun Dinner    4
6      25.29 4.71   Male     No Sun Dinner    4
```

现在，我们尝试来了解变量 tip 的更多信息。首先，小费是怎样分布的呢？我们可以绘制 tip 的密度图，代码如下：

```
# 图15-1a
library(reshape2)
```

```
attach(tips)
par(mfrow = c(3,2))
plot(density(tip),
  main = "a. Density(tip)",
  col = "blue",
  lwd = 2)
```

如图 15-1a 所示，分布呈偏斜状，右侧有一条长长的尾巴。换句话说，一些顾客给的小费相对较多，但大多数顾客的小费都在 2~4 美元左右。这一点很重要，因为许多统计分析方法都依赖于数据至少近似正态分布，或几乎与钟形曲线一致。



小费的多少通常依赖于账单上的消费金额。本章结尾的习题部分会尝试探讨小费和消费金额的关系。

图 15-1：小费（及小费的转换）与正态分布的比较

为了更好地了解变量 tip 和正态分布的差异，我们可以在同一张图上绘制正态分布和 tip 数据。要实现这个操作，可以使用 rnorm() 函数从一个正态分布生成一个大的数字样本，该样本被称作 ran。然后，绘制 ran 的密度图，并用 polygon() 函数填充曲线，将曲线和 tip 密度图区别开来。最后，使用 lines() 函数在同一张图上绘制 tip 的密度图。代码如下。

```
# 图15-1b
ran = rnorm(1000000) # 从规范距离随机采样100万obs
plot(density(ran),
  main = "b. Density(tip) vs. Normal Distribution",
  xlim = c(-4,10))
polygon(density(ran), col = "burlywood")
lines(density(tip),
  col = "blue",
  lwd = 2)
```



图的高宽比（aspect ratio）等于图的高度除以宽度。这个比值很重要，因为改变该比值可以使得图中变量之间的关系更易或更难察觉。感知心理学的研究已经表明，斜率接近 45° 的线效果最佳。选择合适的高宽比以实现最佳效果被称为“倾斜”（banking）。通常，R 散点图会对每个页面采用效果最好的高宽比，不过高宽比也可以通过 asp、mar 和 ylim 等参数进行调整。在本书中，有很多页面有多个图形。显示为 2×2 或 3×3 的正方形时，一个页面上各种图形的高宽比与其在单页图上的比值大致相同。（你可以自己试试看。）图 15-1 的尺寸是 3×2，已经发生了一些变化。在这种情况下，为方便比较，在一个页面上显示 6 张图比保持高宽比更重要。人们在这个问题上意见可能不一致，因此改变高宽比的时候要谨慎。

图 15-1b 为叠加在正态分布上的 tip 密度图，提供的信息比图 15-1a 更丰富，但如果重叠得更紧密一点，评估差异会更容易。也就是说，如果有相同的平均值，这两张图会更容易进行比较。向量 ran 是用 rnorm() 函数的默认选项创建的，平均值为 0，标准差为 1。统计人员经常将变量标准化（standardize）或规范化（normalize），使其易于与已知特征的分布进行比较。

变量 tip 可以通过一个简单的过程标准化，使其均值为 0，标准差为 1。首先，求 tip 的均值和标准差，代码如下：

```
> mean(tip)
[1] 2.998279
>sd(tip)
[1] 1.383638
```

下一步，从每个小费值中减去 2.998（均值），并用所得数据除以 1.384（标准差），创建一个新变量 newtip。这是变量变换（transformation）的一个例子。变量变换是通过函数进行

的，可以在保留原始变量基本信息的同时使新的变量更易于处理，或更符合所用统计方法的必要假设。创建变量 newtip 的代码如下：

```
> newtip = (tip-2.998)/1.384 # 变换tip使得
# mean = 0 且 sd = 1
```

然后，再次绘制这两个密度图，代码如下：

```
# 图15-1c
newtip = (tip-2.998)/1.384 # 变换tip使得
# mean=0 且 sd=1
plot(density(ran),
  ylim = c(0,.48),
  main = "c. Density(newtip) vs. Normal Distribution",
  xlim = c(-4,8))
polygon(density(ran),
  col = "burlywood")
lines(density(newtip),
  col = "blue",
  lwd = 2)
```

图 15-1c 展示了两个密度图的叠加，提供了更多信息。

还有其他方法可以比较两者的分布，会生成不同类型的图形。首先考虑变量 tip 的数值摘要，如下所示：

```
> summary(tip)
   Min. 1st Qu. Median  Mean  3rd Qu.   Max.
  1.000   2.000  2.900  2.998   3.562  10.000
```

summary() 函数给出了第 1 章详细介绍过的分位数——第 25 个、第 50 个（即中位数）和第 75 个百分位数。我们可以根据自己的需要对变量 tip 的分布进行分组，不一定要分成 4 组。例如，你可以将变量分为 10 个百分点一组，断点称为分位数（quantile）。quantile() 函数可以根据用户需求为给定的变量计算分位数，我们来看看它是如何工作的。创建一个新的变量 qtip，其中包含 tip 的分位数，每 10 个百分点分开。分位数必须落在 0~1 的范围内，因此需要使用 seq() 函数，把第一个和最后一个点指定为 0 和 1，并且间隔为 0.1。然后，打印 qtip 的值。其实现代码如下：

```
> qtip = quantile(tip, seq(0,1,.1))
> qtip
 0%     10%    20%    30%    40%
 1.000  1.500  2.000  2.000  2.476
 50%    60%    70%    80%
 2.900  3.016  3.480  4.000
 90%    100%
 5.000  10.000
```

我们可以绘制变量 tip 的分位数和 ran 的分位数，由此确定两个分布的契合程度。例如，把 qtip 的第 10 分位数的值和 ran 的第 10 分位数的值绘在一起。这类图被称为分位数 – 分

位数图（quantile-quantile plot）或 QQ 图，通常易于阅读，因为比较一组点离直线有多近，相较于比较两条曲线更直接。图 15-1d 使用了 qtip2() 函数，使分位数间的间隔更小，这样可以在图上创建更多的点。用 qqplot() 函数绘图，并用 qqline() 函数添加参考线。最后，添加网格以方便读取轴。生成图 15-1d 的代码如下：

```
# 图15-1d
qtip2 = quantile(tip, seq(0,1,.005))
qplot(ran, qtip2,
  main = "d. QQ plot(qtip2)",
  xlim = c(-3,3),
  col = "skyblue2")
qqline(qtip2,
  col = "burlywood",
  lwd = 2)
grid(lty = "dotted",
  col = "gray75")
# 首先需要计算ran和qtip2
```

由图 15-1d 可知，即使 tip 在其大范围内接近正态分布（注意直线），但在两端差得很远，尤其是右侧。这意味着 tip 并不接近正态分布，根据这样的分布进行分析是不明智的。

该图是 QQ 图的示例。现在你对这类图有了一定的了解，你最好知道，事实上不用先创建分位数变量（即 tan 和 qtip）就可以更方便地绘制出几乎相同的图。下面代码中的 qqnorm() 函数可以直接操作变量 tip，生成图 15-1e：

```
# 图15-1e
qqnorm(tip,
  main = "e. Easier way to get QQ plot",
  col = "blue",
  ylab = "tip quantiles")
qqline(tip,
  col = "burlywood",
  lwd = 2)
grid(lty = "dotted",
  col = "gray75")
```

tip 似乎不是正态分布，所以我们需要考虑对 tip 进行变换。回想一下变换的实际意义：如果原始数据不满足分析给出有效结果的必要假设，有时可以应用数据函数（即变换）产生满足假设的数据，然后对变换后的数据进行分析。这样一来，结论必然是关于变换后的数据，而不是原始数据的。这里有许多方法可以尝试，但对于减少或消除偏斜（分布非对称的程度），对数函数的变换通常是成功的。下面的代码生成了图 15-1f，由此可知，对数（普通或基于 10）对数据的变换效果非常好：

```
# 图15-1f
logtip = log10(tip)
qqnorm(logtip,
  main = "f. QQ plot of log10(tip)",
```

```
  col = "blue4")
qqline(logtip,
  col = "burlywood3",
  lwd = 2)
```

目前为止，我们已经使用 QQ 图对一个变量的分布及其理论分布进行了比较。你也可以用 QQ 图来比较两个变量的分布，比如 tips 数据集中的 tip 和 size。对于这个数据集来说，比较男性顾客和女性顾客给的小费，或午餐和晚餐时给的小费等，可能更有趣。要实现这个需求，你可以组织适当的数据子集，并绘制各组的 QQ 图。当变量刚好有两个级别时（比如 tips 数据集的性别），用 lattice 包比较两组较为便捷。你可以使用如下 qq 函数：

```
qq(y ~ x)
```

本例中，*y* 正好有两个等级，并且 *x* 是定量变量，代码如下：

```
# 图15-2(左)
library(lattice)
qq(sex ~ tip,
  main = "Tips given by men and women")
```

结果如图 15-2（左）所示。

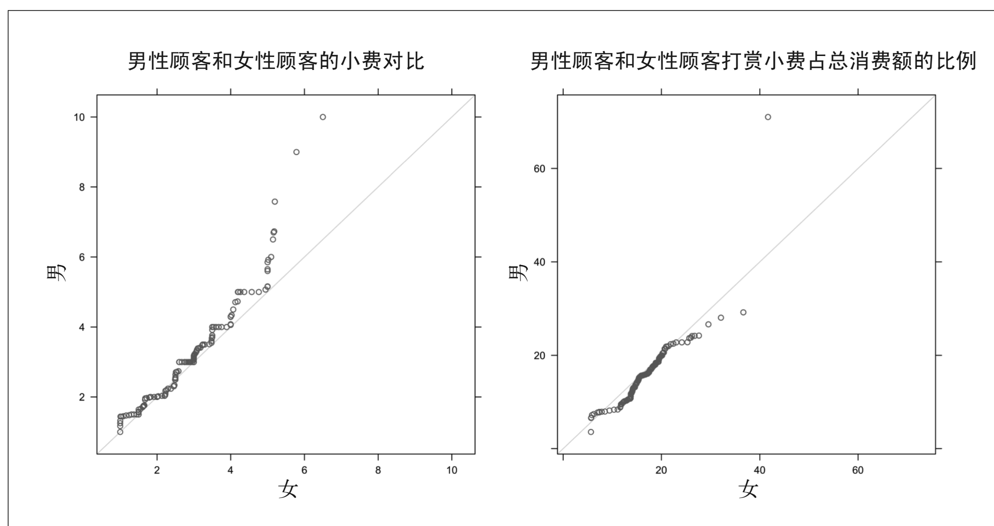


图 15-2：男性顾客和女性顾客的小费对比 QQ 图（左），以及男性顾客和女性顾客打赏小费占总消费额比例的 QQ 图（右）。两张图均使用 lattice 包生成

在图 15-2 中，左侧图显示了男性顾客和女性顾客的小费分布。在图中，5 美元以下的情形看不出什么差异，但是高于 5 美元的小费更有可能是从男性顾客那里获得的。这一点可以用数值摘要来证明，代码如下：

```
> summary(tip[sex == "Male"]) # 仅包含男性的子集
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    2.00    3.00    3.09    3.76  10.00
> summary(tip[sex == "Female"]) # 仅包含女性的子集
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   2.750   2.833   3.500   6.500
```

通过研究小费的分布，我们发现了一些有趣的现象。相比之下，研究小费和账单总额之比可能更有意义。毕竟，关于打赏的指导大多建议按照"账单的 15%"给小费。把小费与账单总额的比例乘以 100，可以计算出一个新的变量。然后，可以按照这个新变量为男性顾客和女性顾客绘制一张 QQ 图，代码如下：

```
# 图15-2(右)
tips$ratio = 100*(tip/total_bill)
qq(sex ~ tips$ratio,
  main = "Tips as percent of total bill, for men and women")
```

在图 15-2 中，右侧图显示在比例高达约 25% 的地方，男性和女性同样慷慨，而越往上，女性越大方。这里只有一个例外。也许这个人想给他的女朋友留下好印象，也许小数点在这里出现了误记。这样的问题在真实的数据分析中经常出现。你需要检查数据是否正确。此外，你需要决定是包括还是排除那样的极值点，或者用两种方式来分析数据，并报告两者的结果。这非常有趣！

练习15-1

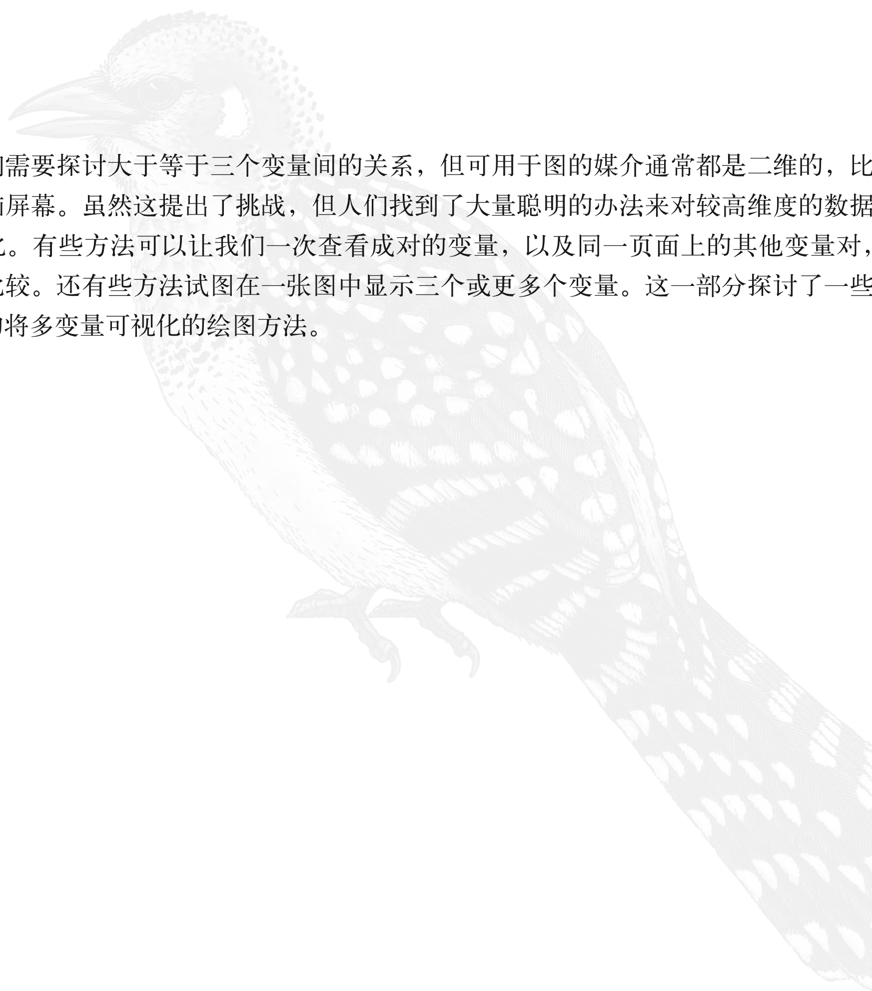
继续分析 tips 数据集。研究变量 ratio，而不是变量 tip。其他因素和小费的多少有关吗？表面上的关系合理吗？其他因素会产生误导吗？

练习15-2

再来看 car 包中的 Vocab 数据集。变量 vocabulary 和 education 符合正态分布吗？男性和女性的 vocabulary 得分的分布相同吗？

第四部分

多变量图



有时，我们需要探讨大于等于三个变量间的关系，但可用于图的媒介通常都是二维的，比如纸或电脑屏幕。虽然这提出了挑战，但人们找到了大量聪明的办法来对较高维度的数据进行可视化。有些方法可以让我们一次查看成对的变量，以及同一页面上的其他变量对，以便进行比较。还有些方法试图在一张图中显示三个或更多个变量。这一部分探讨了一些前途无量的将多变量可视化的绘图方法。

第 16 章

散点图矩阵和相关性分析图

16.1　散点图矩阵

当面对许多定量变量时，首先查看每个可能的变量对之间的关系有时会有帮助。为了避免为每个组合都输入 plot() 命令，R 提供了一个快捷命令 pairs()，它可以执行这一操作。此外，pairs() 将所有图放在同一页面上，以便进行比较。这样得到的图被称为散点图矩阵（scatter plot matrix）。我们将使用散点图矩阵来研究各种教会团体成员性格之间的关系。

对美国宗教生活的长期研究表明，每周出勤情况以及成员的关系似乎与教会的严格程度相关。Iannaccone（1994）讨论了这项研究，并给出了一个有趣的数据集，该数据集为 18 个宗教教派中的每一个显示了几个变量。从 Sleuth2 包的 ex1713 中可以找到该数据。代码如下：

```
> library(Sleuth2)
> attach(ex1713)
> head(ex1713)

        Denomination Distinct Attend NonChurch StrongPct AnnInc
1      American Baptist    2.5   25.6      1.01      50.6  24000
2    Assemblies of God    4.8   35.4      0.68      58.6  27100
3             Catholic    3.0   26.4      1.43      40.0  32900
4  Disciples of Christ    2.1   24.3      2.58      47.0  28600
5            Episcopal    1.1   17.3      1.93      32.0  39000
6 Evangelical Lutheran    2.7   23.0      1.71      41.5  33700
```

要查看这些数据的码本，请输入如下命令：

```
> ?ex1713
```

码本简介如下：

`Distinct`

　纪律的特殊性 / 规范性，7 分制。

`Attend`

　每周出勤的平均百分比。

`NonChurch`

　成员所属的世俗组织的平均数量。

`StrongPct`

　认为自己是强大的教会成员的人数的平均百分比。

`AnnInc`

　年平均收入。

图 16-1 所示的散点图矩阵是由 `pairs()` 函数生成的。注意，变量名输入是一个公式，从符号“~”开始，紧随其后的就是变量名。变量名按出现在图上的顺序排列，用符号“+”分隔。此外，可以为该函数添加任意特殊参数，以及 `par()` 参数。生成图 16-1 的代码如下，这里只使用了参数 `pch` 和 `col`：

```
# 图16-1:生成教会宗派数据的散点图矩阵
library(Sleuth2)
attach(ex1713)
pairs(~ Distinct + Attend + NonChurch + StrongPct + AnnInc,
pch = 16,
col = "deepskyblue")
```

图 16-1 中的散点图矩阵对任意两个变量都进行了两次绘制。在每个组合中，每一个变量都做一次 *x* 变量，一次 *y* 变量。例如，第二行的变量 `Attend` 是四张散点图的 *y* 变量，并且其他四个变量分别是四张图的 *x* 变量。在第二列，`Attend` 是四张散点图的 *x* 变量，并且其他四个变量分别是四张图的 *y* 变量。

在第二行我们可以看到，`Attend` 和 `Distinct` 正相关。也就是说，其中一个变量增加，另一个变量也随之增加。同样，`Attend` 和 `StrongPct` 之间也是正相关的。而 `Attend` 同 `NonChurch` 和 `AnnInc` 则为负相关，即一个变量增加时，另一个变量减小。负相关不如正相关那么明显。换句话说，呈现参数负相关的点并没不像呈现正相关的点那样紧紧地环绕在一条直线周围。这一点在图 16-3 中更为明显，图中每一张散点图都有一条最小二乘直线。当然，强相关也并不一定意味着因果关系。换句话说，即使高严格性和高出勤率通常同时出现，也不能证明二者之间存在必然联系。不过，这确实表示它们之间的联系可能十分有趣，值得进一步研究。

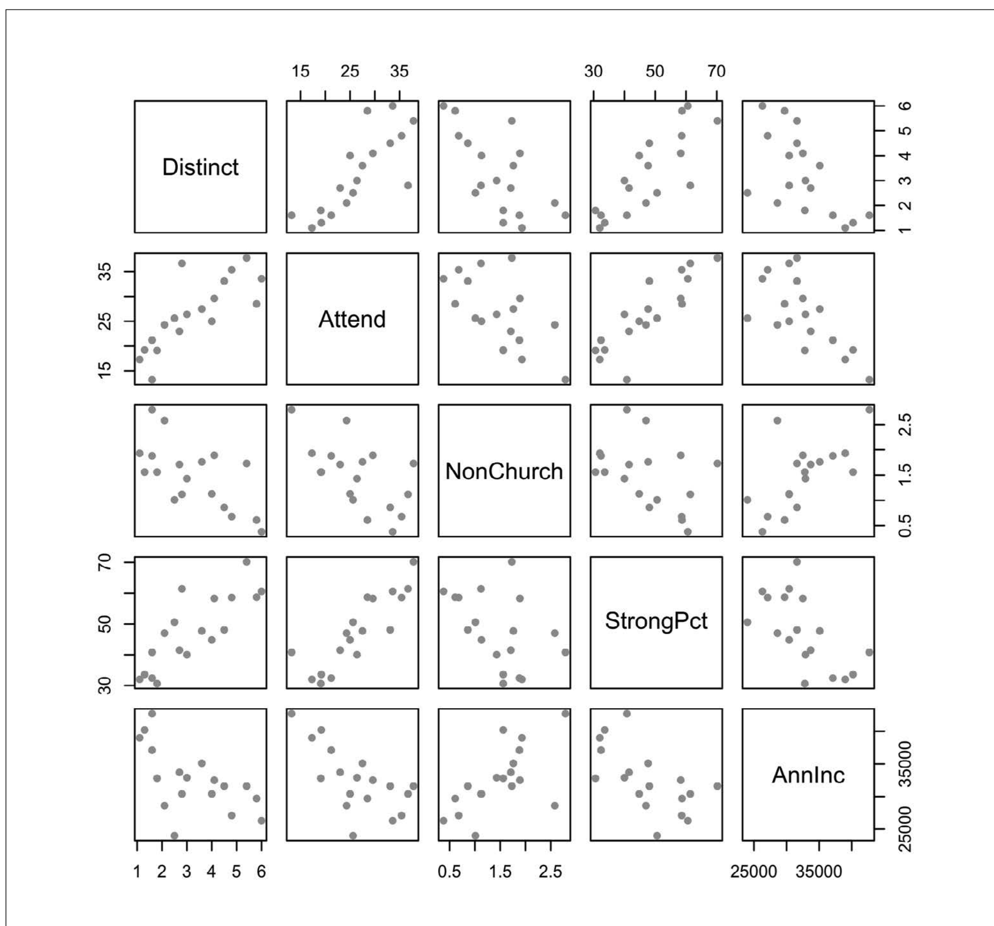


图 16-1：教会宗派数据的散点图矩阵

car 包中有一个叫作 scatterplotMatrix() 的函数，它为散点图矩阵添加一些有用的特性。首先，它可以让你在矩阵对角线上轻松绘制每个变量的分布，比如绘制直方图、密度图、箱线图、QQ 图或一维（对角线的）带状图。此外，它还可以为每张图添加最小二乘线。

平滑器（smoother）也可以在每一张图上应用。正如在第 12 章提到的，平滑器可以使散点图中的模式易于观看。几种平滑器在给定 x 值（或相近的几个 x 值）处都显示了 y 的中心，这种方式使连接这些点构成的线（通常是曲线）相对平滑。图 16-2 显示了应用平滑器之后的散点图矩阵，平滑器用一条红线表示。你可以使用参数 smoother 选择一种平滑方法，图 16-2 使用了默认方法，即“loess”或局部加权回归（locally weighted regression）。

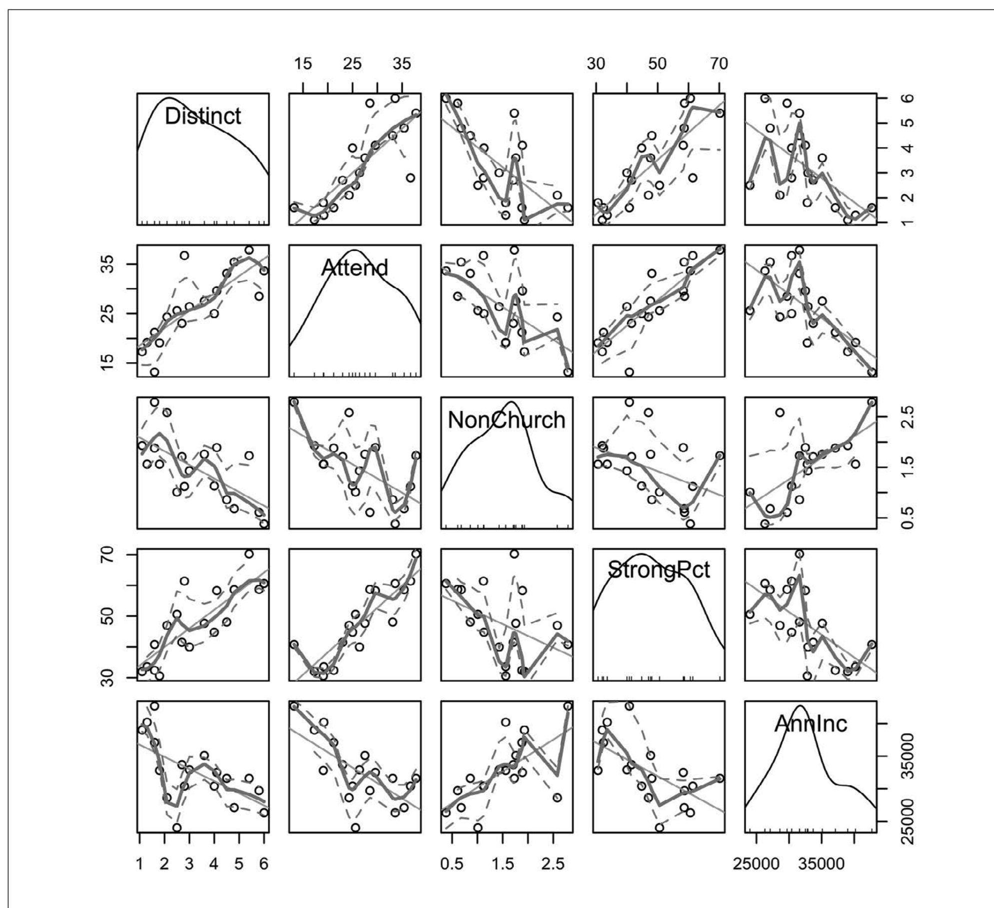


图 16-2：通过 car 包中的 scatterplotMatrix() 函数生成的散点图矩阵。默认选项在对角线添加核密度图和地毯图，在每张图的窗口内添加最小二乘线和平滑器

生成图 16-2 的代码如下：

```
# 图16-2:散点图矩阵w/平滑和对角线密度
library(car)
library(Sleuth2)
attach(ex1713)
scatterplotMatrix(~Distinct + Attend + NonChurch + StrongPct +
  AnnInc)
```

图 16-2 中由平滑器生成的线揭示了几个有趣的现象。Attend 和 Distinct 之间的联系几乎可以表示为一条直线，Attend 和 StrongPct 也是这样，也就是说这类关系可以描述为简单的线性相关。在简单的散点图上看似接近线性的关系（例如 Attend 和 AnnInc 的关系），在这张图中显得更为复杂。然而，应当指出的是，该数据集中只有 18 个教派，对于得出任意两个变量之间的关系来说，这个数量相当小。这个例子只是为了说明包中可

用的功能。大多数情况下，绘制图 16-1 可能就够了。对数据有了感受，你或许会发现其他实用的特征。

scatterplot() 生成的矩阵可以进行定制。使用参数 smoother = NULL 可以忽略平滑器，代码如下所示。类似地，使用参数 reg.line = F 可以去掉回归线，使用参数 diagonal 可以改变对角线处的图的类型。要查看选项，请输入 **?scatterplotMatrix**。

定制的 scatterplot() 矩阵图如图 16-3 所示，创建代码如下：

```
# 图16-3:散点图矩阵w/out 平滑且用直方图表示
scatterplotMatrix(~Distinct + Attend + NonChurch + StrongPct
  + AnnInc, diagonal = "histogram",
  smoother = NULL)
```

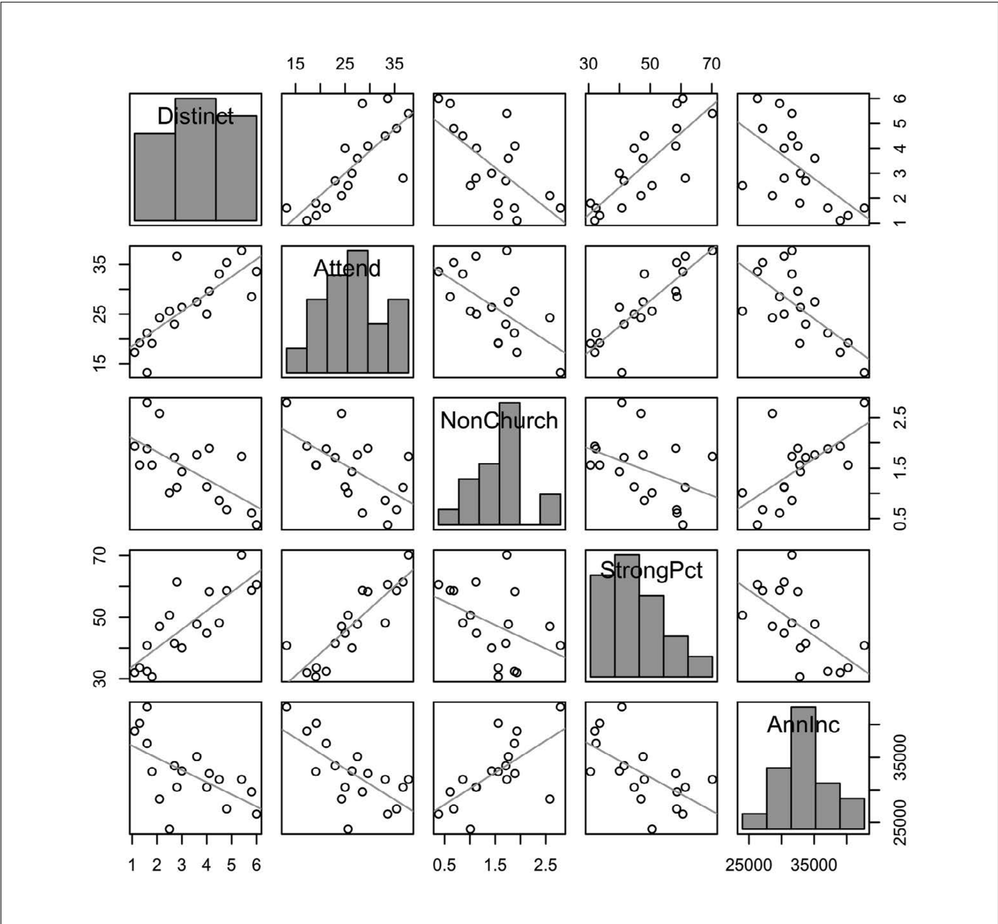


图 16-3：由 car 包中的 scatterplotMatrix() 函数生成的散点图矩阵。去掉了平滑器且用直方图取代了对角线的密度图

图 16-3 展示了对角线为直方图的矩阵。相比默认生成的密度图，这可能是一个更好的选择，至少在本例中是这样，因为样本大小只有 18。你可能认为，变量对 `Attend` 和 `NonChurch` 的分布没有密度图上的光滑。此外，`NonChurch` 的两个特别大的值会导致此变量与 `Attend` 之间的关系比实际情况更强、更线性。仔细观察这两个变量的散点图，你就会发现这一点，但如果没有用直方图进行标记，你可能就无法发现这一点。

研究散点图矩阵时一定要记住，它实际上为你呈现了许多独立的图。不要被页面上的大量信息量所吓到，单独看每一张图就好。看了许多图以后，你就会发现进行对比会很有启发。

16.2　相关性分析图

相关性分析图（corrgram，有时也称为 correlogram，虽然实际上此术语指的是其他东西）是一种和散点图矩阵相关的图。在这种类型的图中，各个散点图被替换为代表数字的符号，这些数字衡量的是两个定量变量之间的线性相关。*皮尔森相关系数*（Pearson correlation coefficient）通常表示为 r，在 –1 和 1 之间变化。完全正相关时，$r = 1$，表明两个定量变量在散点图上的所有点正好在一条上升的直线上。完全负相关时，$r = -1$，表明所有的点正好在一条下降的直线上。r 接近 0，表示变量之间关联很少或没有关联。注意，相关系数衡量的不是直线的倾斜度，而是测量点到直线的总偏差。图 16-4 举例说明了相关系数的含义。请注意：只有当变量之间是线性关系时，也就是说点落在直线上时，相关系数才是有用的。在其他情况下，相关系数可能产生误导，甚至是欺骗性的。

要绘制相关性分析图，首先必须定义一个*相关矩阵*（correlation matrix），该矩阵包含数据集中所有变量对的相关系数，由 `cor()` 函数实现，代码如下：

```
> library(Sleuth2)
> attach(ex1713)
> y = cor(ex1713[, 2:6]) # 使用全部的第2~6行与列
> y
            Distinct     Attend  NonChurch  StrongPct     AnnInc
Distinct   1.0000000  0.7891067 -0.6585883  0.8127124 -0.6003892
Attend     0.7891067  1.0000000 -0.6107342  0.8649691 -0.6766143
NonChurch -0.6585883 -0.6107342  1.0000000 -0.4218525  0.6458747
StrongPct  0.8127124  0.8649691 -0.4218525  1.0000000 -0.6146261
AnnInc    -0.6003892 -0.6766143  0.6458747 -0.6146261  1.0000000
```

相关矩阵生成后，可以使用 `corrplot` 包中的 `corrplot()` 函数生成几种类型的相关性分析图。图 16-5 展示了一些例子，所有例子都使用颜色来描述相关性大小。你也可以使用对象的大小、对象的方向，或数字显示给定变量对的相关性程度。

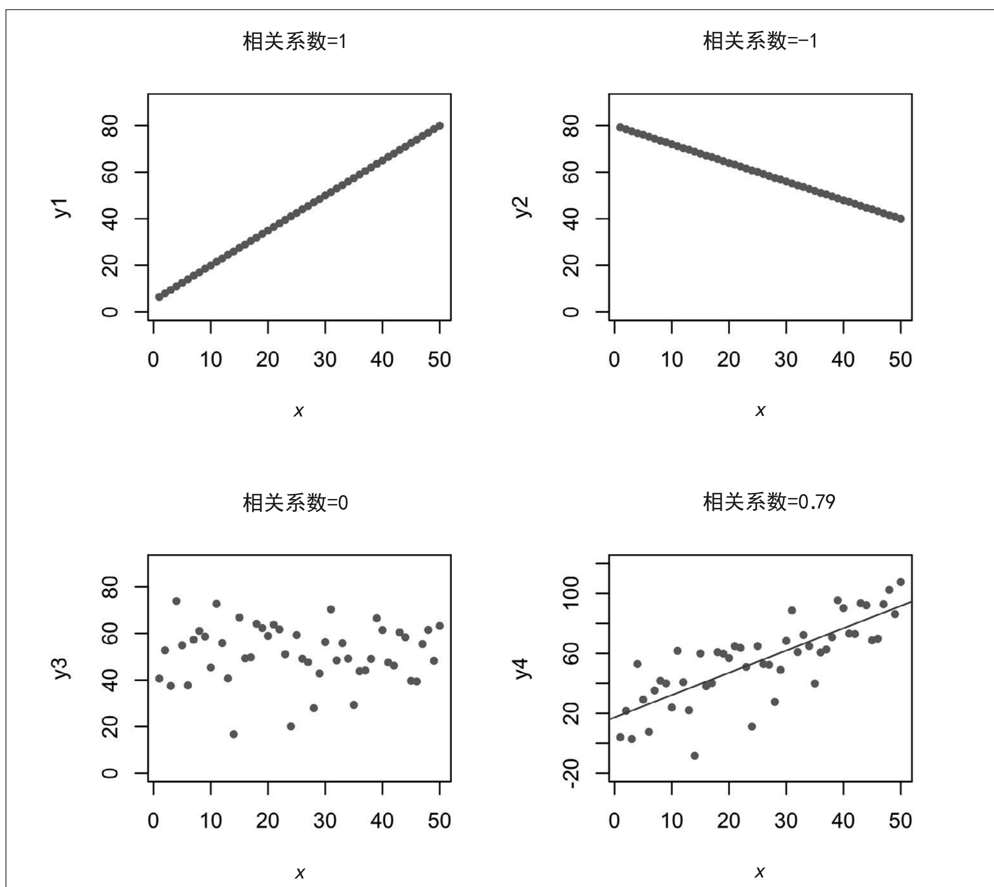


图 16-4：相关系数为 1，则两个变量完全正相关，所有点正好落在向上倾斜的直线上。相关系数为 –1，则两个变量完全负相关，所有点落在向下倾斜的直线上。相关系数为 0，表示没有明显的关联模式。相关系数为 0.79，表示点“接近”一条直线

变量 A 和 B 之间的关系与 B 和 A 之间的关系相同，因此完整的相关性分析图是多余的。也就是说，上半部分的相关性与下半部分的相关性完全相同。因此，有些人更倾向于只显示上半部分或下半部分矩阵。图 16-5 中右下角的图就是一个例子。你可以使用参数 `type = "lower"` 实现此操作。生成如图 16-5 所示的相关性分析图的代码如下：

```
# 图16-5:多种相关性分析图
library(corrplot)
library(Sleuth2)
attach(ex1713)
y = cor(ex1713[, 2:6])
par(mfrow = c(2,2))
corrplot(y)  # method的默认值为"circle"
corrplot(y, method = "color")
corrplot(y, method = "number")
corrplot(y,  method = "ellipse", type = "lower")
```

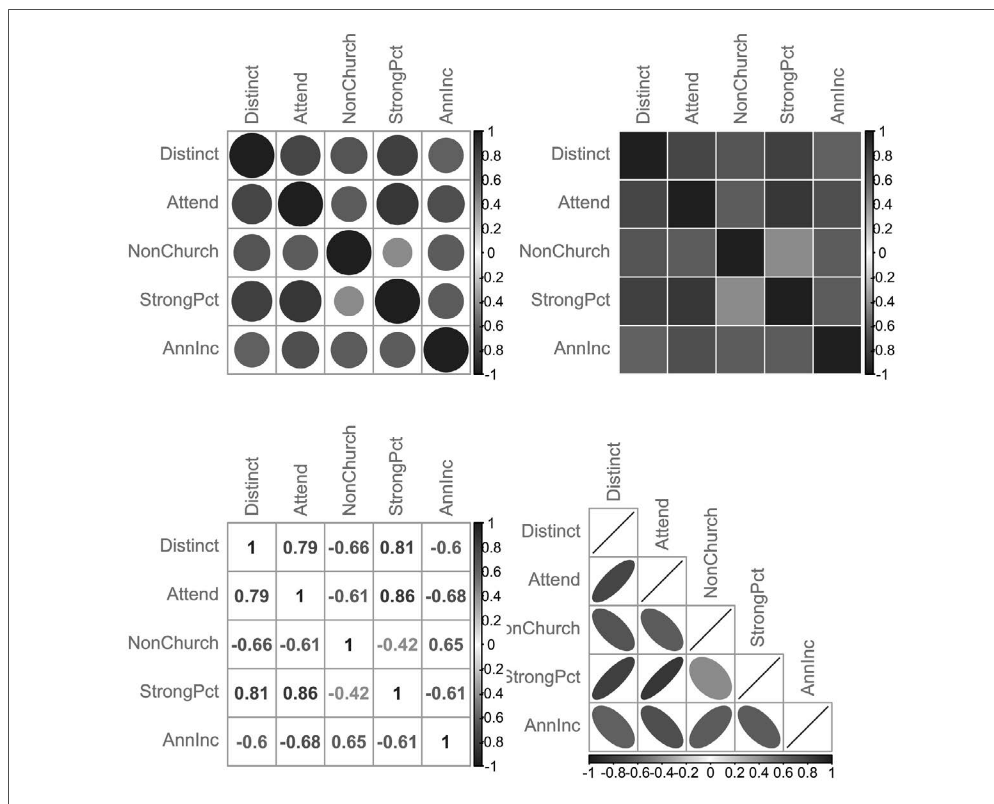


图 16-5：相关矩阵的可视化。这是一种概括或近似的散点图矩阵，由 `corrplot` 包中的 `corrplot()` 函数生成。左上图，`method = "circle"`；右上图，`method = "color"`；左下图，`method= "number"`；右下图，`method= "ellipse", type="lower"`（另见彩插）

尽管有人对相关系数的缺点发出过警告，但相关性分析图仍是呈现数据的有效方式，只要你能花些时间观察散点图（或平滑器），先看看相关系数在图中是否有意义。比较图 16-5 中的相关性分析图和本章前面的散点图矩阵，看看从这些不同的展示中得出的结论是否一致。用 `psych` 包中的 `cor.plot()` 函数也可以绘制相关性分析图。

图 16-5 中的所有图都用颜色表示关联强度，在右边或底部以渐变顺序列出各种颜色的含义。蓝色阴影呈正相关，颜色越深关联越强（即接近 1）。红色阴影呈负相关，颜色越深越接近 –1。（左上角和右下角）两张图中，图形大小也表示关联强度，但表示方式相反。在左上角的图中，较大的图形表示较大的绝对值。在右下角的图中，方向指示正相关或负相关，窄椭圆形表示点紧挨着线即强相关。宽椭圆形表明线周围有很多震动，即弱相关。无需多言，你也许认识到了这一点，经过确认就更好了，对吗？

把一部分图放在矩阵的下半部，另一部分放在上半部，也可将散点图矩阵与相关分析图相结合。`GGally` 包中的 `ggscatmat()` 函数可以实现这一点，代码如下：

```
# 图16-6
library(GGally)
library(Sleuth2)
ggscatmat(ex1713, columns = 2:6)
```

结果如图 16-6 所示。

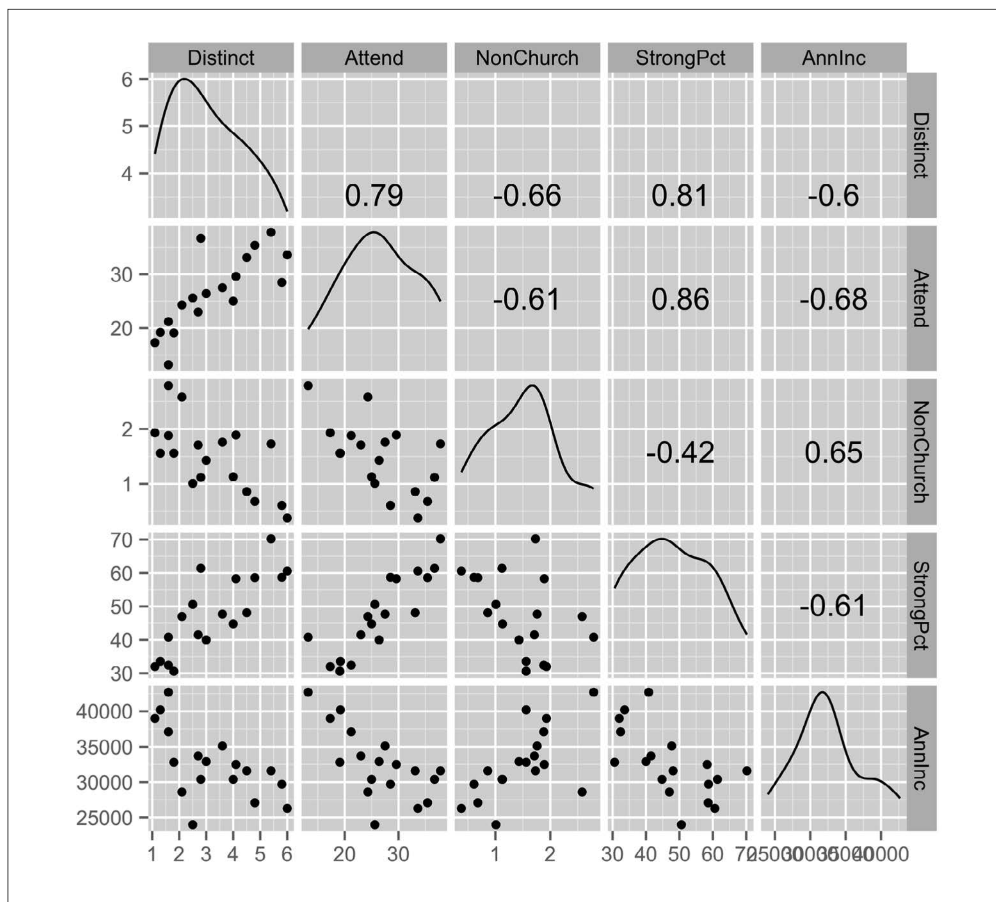


图 16-6：散点图矩阵和相关性分析图的结合，由 GGally 包中的 ggscatmat() 函数生成。注意右下角的 x 轴有叠加，这可以处理掉

注意，图 16-6 中出现了一个小问题。在右下角，x 轴的值重叠了，这是因为数字太大，而空间太小。对此，有一个非常简单的解决方案，将 AnnInc 值的范围从几美元变为数千美元，并用新变量重新绘制图即可。为此，需要一条新命令，还需要对另一个命令稍做改变。首先，创建一个新变量 Inc，等于 AnnInc 除以 1000。新变量成为数据阵中的第七列。下一步，修改 ggscatmat() 命令，使其包括所需的列，即去掉 AnnInc 而包括 Inc，代码如下：

```
# 图16-7:解决图16-6中的小问题
library(GGally)
library(Sleuth2)
ex1713$Inc = ex1713$AnnInc/1000
ggscatmat(ex1713, columns = c(2:5,7))
```

请看图 16-7 所示的结果，观察叠加问题是如何解决的。

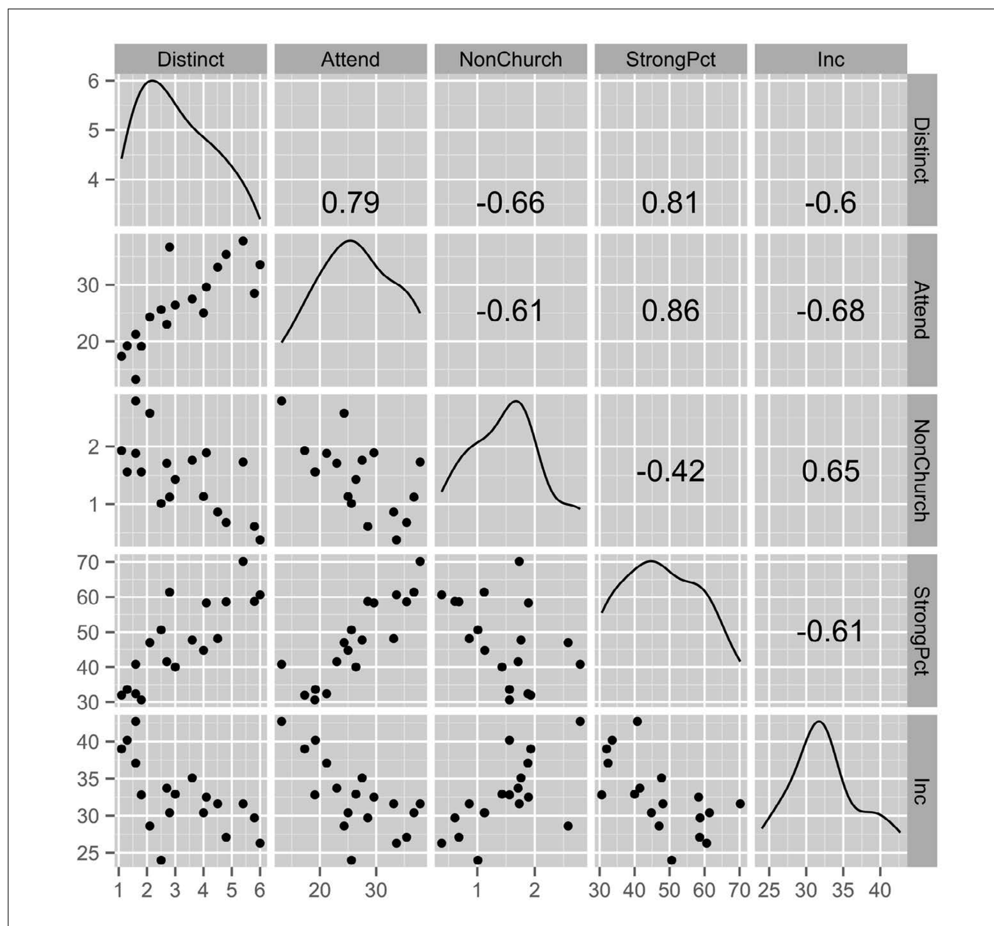


图 16-7：图 16-6 中的问题已修复。注意，现在右下角的 *x* 轴的可读性增强了

16.3 混合定量变量和分类变量的广义对矩阵

既有定量变量又有分类变量的数据集是很常见的。在这种情况下，虽然散点图不能处理分类变量，但仍然可以以有意义的方式展示所有变量对。这意味着仅显示几种类型的图，每一种和相关变量类型相适应。如下代码示例用 `GGally` 包中的 `ggpairs()` 和 `gpairs` 包中的 `gpairs()` 展示了此类图形化展示。

再次考虑 Nimrod 数据集。该数据集只有一个定量变量 time，以及两个分类变量 level 和 medium。变量 performer 只是名字，不会提供任何有帮助的信息，还会使页面更拥挤，所以将其舍去。使用数据集的子集可以实现这一操作（详见 12.1 节）。我们需要的子集是 Nimrod[, 2:4]，即只包含第二列到第四列的所有行的数据，代码如下：

```
# 图16-8
library(GGally)
ggpairs(Nimrod[,2:4])
```

结果如图 16-8 所示。

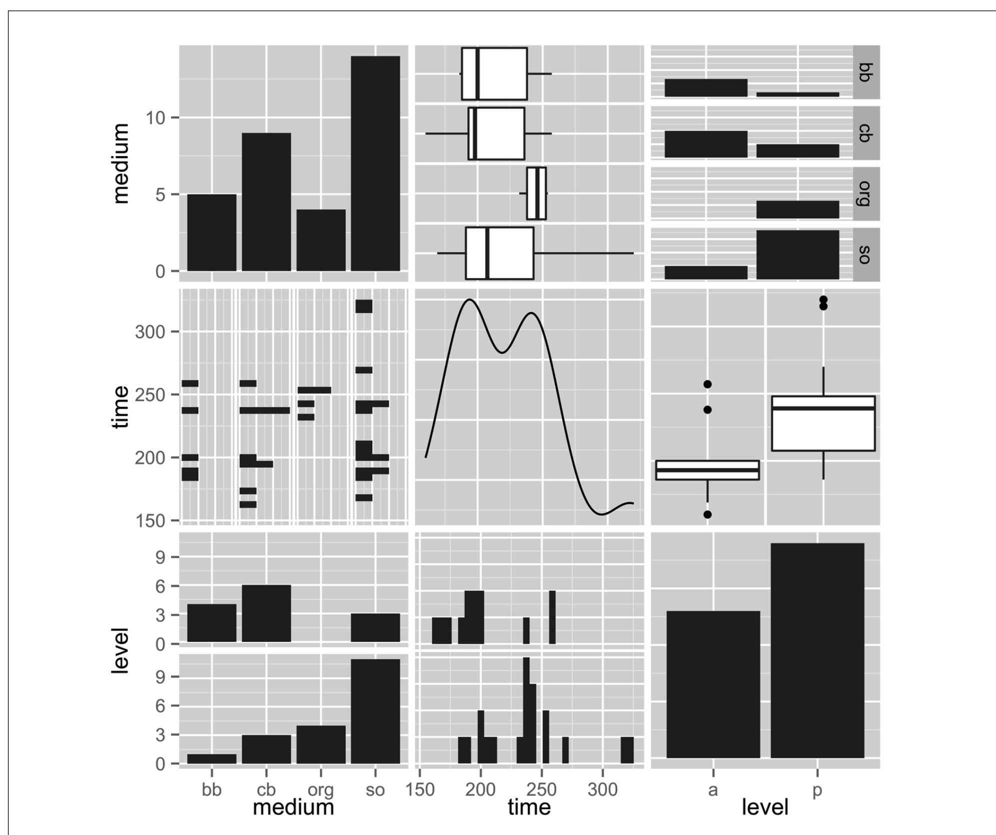


图 16-8：广义对矩阵，用于处理混合定量变量和分类变量的数据，通过 GGally 包中的 ggpairs() 生成

图 16-8 中有一些类型的图我们比较熟悉。对角显示了两个分类变量的条形图以及一个定量变量的密度图。还有几张箱线图，其中每一对变量包括一个定量变量和一个分类变量。

此外，还有几张我们还没提到过的图。左下角和右上角是同样两个分类变量的图。在这里，通过方格分解成了多个条形图。左下角是 medium 的条形图，每一个 medium 对应两个

level 值。右上角是对应每个 medium 值的 level 条形图。

最后，一些方块内绘制了分类变量和定量变量的条形码图（barcode plot）。例如，在左边中间方块内呈现的数据，看起来像印在书的封面或其他商品上的条形码。每一个小条表示一个点，排列得如同四条带状图，每个 medium 值对应一条。有联系时，第二条不是简单的叠加，而是放在该位置已有的条的旁边。换句话说，这些条是跳动的，但井然有序。最后一行中间的图也是条形码图，但这张图包含了对应业余和专业的两个带状图。

关于该图，还有最后一点值得观察。请注意，有一个定量变量和一个分类变量时，这对变量会以两种不同类型的图展示出来。关于两个变量的关系，从两张图中得出的结论也许略有不同。这里有许多可供选择的选项。更多信息，请输入 **?ggpairs** 查询。

如下代码中的 gpairs() 函数也给出两两比较的视图，且引入了另一种类型的图——马赛克图（mosaic plot）。第 20 章会专门介绍这类图，所以这里不会详细讨论。了解马赛克图后，回顾本例，再次比较图 16-8 和图 16-9 也许会让你有所收获。生成图 16-9 的代码如下：

```
# 图16-9
install.packages("gpairs") # 若还没安装
library(gpairs)
gpairs(Nimrod[,2:4])
```

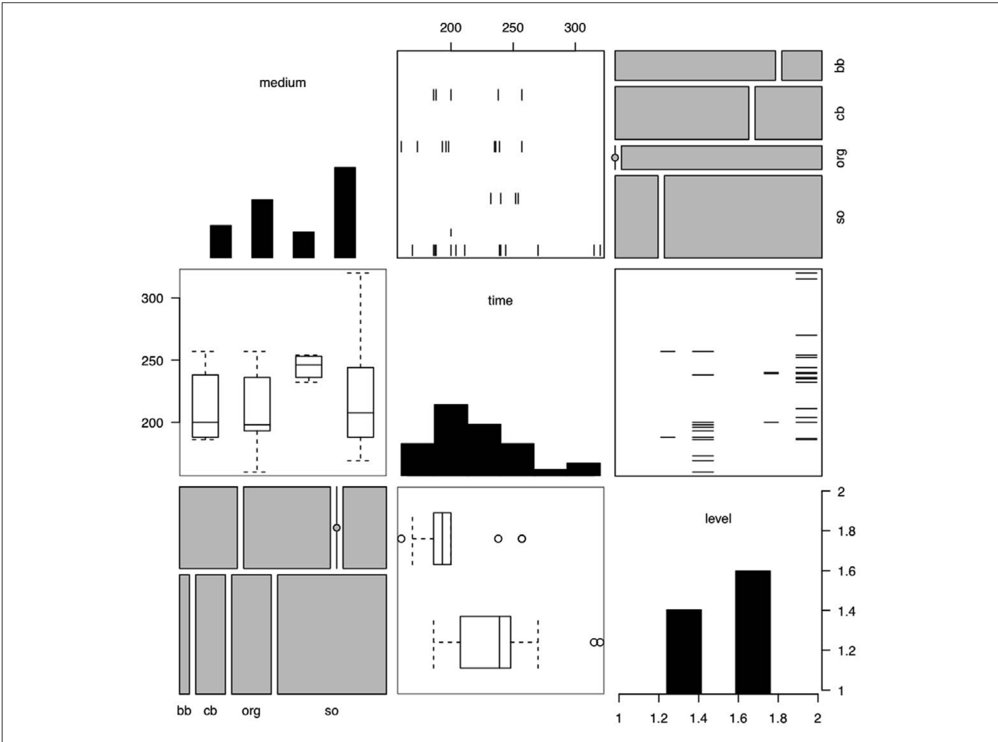


图 16-9：广义对矩阵，用于定量变量和分类变量混合的数据。使用 gpairs 包中的 gpairs() 生成

仔细观察图 16-9，对角线上有条形图和直方图。它们分别是哪一张图？为什么？如果你不确定，复习第 7 章和第 9 章。查看 `gpairs()` 有哪些可用选项，请输入 **`?gpairs`**。

练习16-1

请使用本章介绍的工具研究 `car` 包中的 `Ginzberg` 数据（仅用前三个变量）。你是否发现了有趣的关系？这里是否存在线性关系？

第17章

三维图

17.1 三维散点图

trees 数据集有三个定量变量。我们用带状图查看了其中一个变量 Volume 的分布，并用散点图查看了两个变量 Height 和 Girth 的关系。散点图有一种扩展图形可以同时可视化三个变量，通常称为三维散点图（3D scatter plot）。很多包都有创建三维散点图的函数，包括 lattice、scatter plot3D、rgl、plot3D、car，等等。

本章重点关注 scatterplot3d 包，因为它的语法和基础 R 中的 plot() 函数非常相似，操作起来相对容易，而且非常灵活。另外，使用该包便于展示许多技巧，而这些技巧可使三维图易于理解。这里还将介绍两三个其他函数，进行对比讨论。

scatterplot3d() 函数的基本语法如下：

```
scatterplot3d(x, optional arguments)
```

其中 x 是数据框或矩阵。scatterplot3d() 函数还可以写为如下形式：

```
scatterplot3d(x, y, z, optional arguments)
```

其中 x、y 和 z 是向量。

虽然第一种形式通常较为方便，但第二种更好，因为它可以让你决定变量的顺序，或者选择变量子集。变量 *x* 绘制在水平轴上，*y* 在对角线上，*z* 在垂直轴上。在下面的示例脚本中，*x*、*y* 和 *z* 分别为 Height（高度）、Girth（周长）和 Volume（体积）：

```
# 图17-1的脚本
library(scatterplot3d)
attach(trees)
par(mfrow = c(2,2),
  cex.main = .9,
  las = 1)

scatterplot3d(Height, Girth, Volume,
main="a. 3D scatter plot of trees data")
# 可以替代为:scatterplot3d(trees)
# 看下会发生什么……

scatterplot3d(Height,Girth,Volume,
  pch = 16,
  highlight.3d = T,
  main = "b. 3D scatter plot with highlighting",
  cex.axis = .5)

scatterplot3d(Height,Girth,Volume,
  pch = 16,
  highlight.3d = T,
  type = "h",
  main = "c. 3D scatter plot with lines and highlighting",
  cex.axis = .5)

scatterplot3d(Height, Girth, Volume,
  pch = 15, type = "h",
  lwd = 5,
  color = "cyan4",
  main = "d. 3D bar plot without box",
  box = F,
  cex.axis = .5)
```

脚本的结果如图 17-1 所示。

图 17-1a 展示了一个基本的三维散点图。盒子底部有网格，对于在二维平面上显示三维空间有一定的帮助。然而，在图中估计给定点的坐标仍然相当困难。

图 17-1b 做出了改进。指定参数 `highlight.3d=T` 为点添加了颜色，“前面”的点（y 值较小的那些点）为灰色，随着 y 值的增加，颜色逐渐变暗。参数 `pch=16` 填充了圆圈，颜色效果由此加强。

图 17-1c 添加了各点到底部网格的垂直线，再次做出改进。为实现这一点使用了参数 `type="h"`，这样更容易准确地辨别 x 和 y 值。

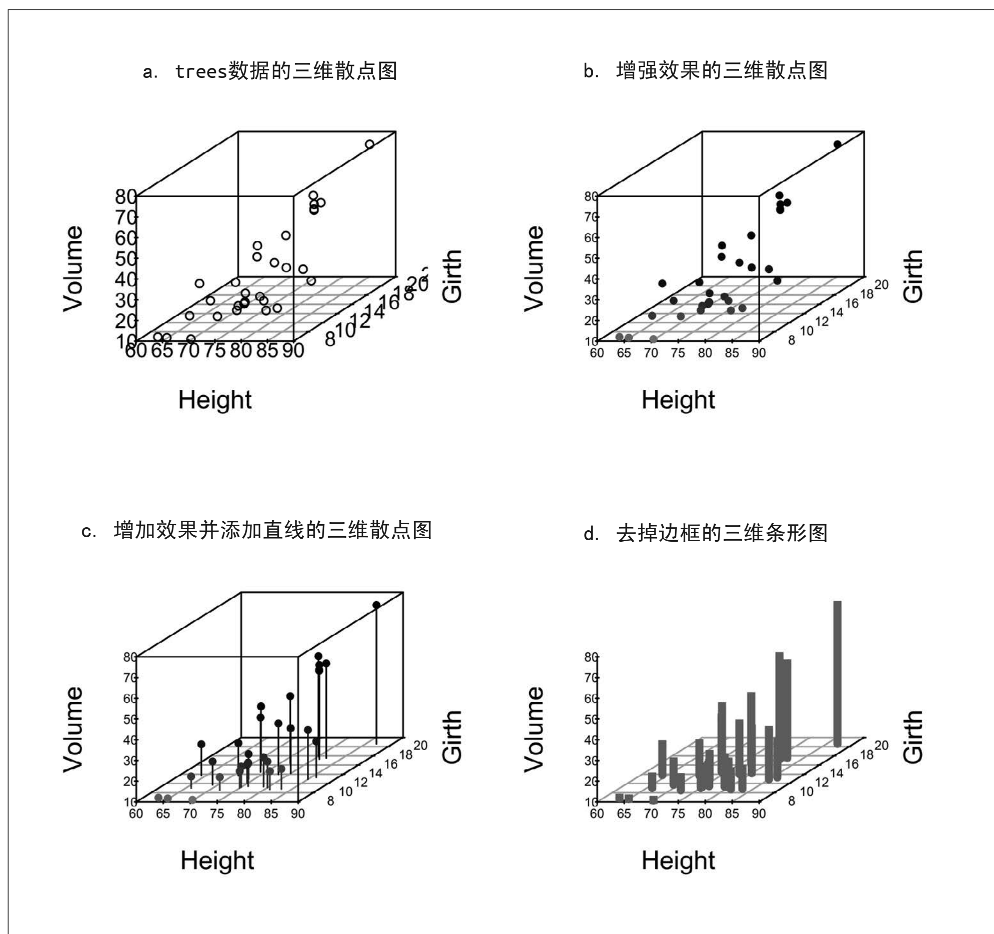


图 17-1：基本 scatterplot3d() 的输出和一些改进

最后，通过把图符变为方形，并增加垂直线的宽度以匹配正方形的宽度，图 17-1d 把线图变成了条形图。使用参数 pch=5 可以改变图符，lwd=5 可以改变线宽，你通常需要几次试错，以找到合适的线宽。使用 box=F 可以删除图周围的方框。判断一下，哪张图最易于阅读？

另一种帮助观众感知三维图的方式是在图上放置一个*参考面*（reference surface），这可以是一个平面或一个曲面。图 17-2 展示了一种可能性，即由一个线性模型定义的预测平面。如果你并不了解多种回归，可以跳过这个例子。

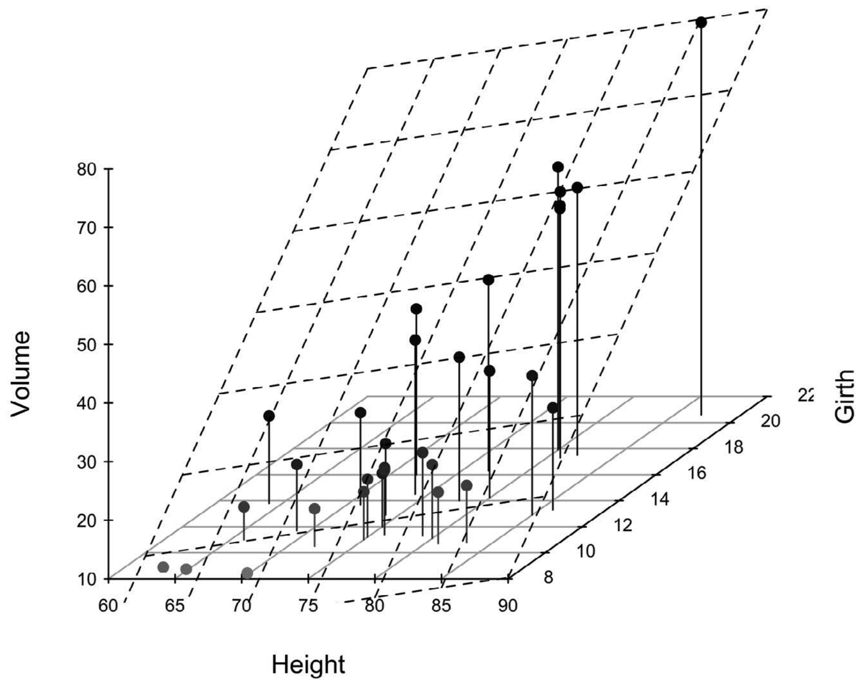


图 17-2：使用 `scatterplot3d()` 绘制的带有预测平面 / 参考面的三维散点图

生成图 17-2 的代码如下：

```
# 图17-2
library(scatterplot3d)
attach(trees)
par(mfrow=c(1,1), las = 1)
# 将图的结果存入对象sp3
sp3 = scatterplot3d(Height, Girth, Volume, pch = 16,
  highlight.3d = T,
  type = "h",
  main = "3D scatter plot with prediction plane",
  cex.axis = .7,
  box= F)
model = lm(Volume ~ Height +Girth) # 拟合线性模型
  "model" sp3$plane(model)         # 绘制model创建的平面
```

图 17-2 有点混乱，因为很难判断特定点是高于还是低于参考面。我们可以通过 `car` 包中的 `scatter3d()` 函数生成更好的图，结果如图 17-3 所示。这张图与前面的图相似，但平面有了颜色，增加了存在感，并且处在平面上方和下方的点绘制成了不同颜色。

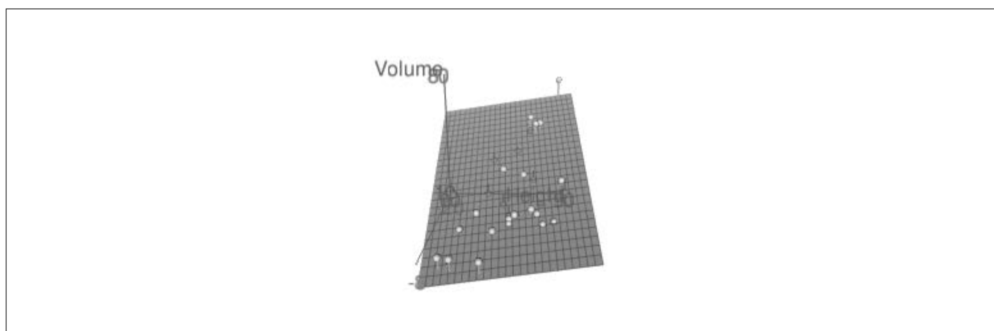


图 17-3：通过使用 `car` 包中的 `scatter3d()` 函数绘制的带有预测面板的三维散点图（另见彩插）

同 `scatterplot3d()` 不一样，`scatter3d()` 把 *y* 作为垂直坐标轴，所以，为了绘制易于和由 `scatterplot3d()` 生成的图进行比较的图，这里必须改变变量的顺序。此外，变量 *z* 的方向是相反的，因此变量 `Girth` 要乘以 –1，使其可与图 17-2 相比较。编写代码之后，你就会完全明白这些做法的意义。以下是生成图 17-3 的代码：

```
# 图17-3
library(car)
library(rgl)
attach(trees)
scatter3d(Height, Volume, -1*Girth)
rgl.snapshot("ch17.3.png", fmt = "png") # 存入工作目录
```

图 17-3 显然更清晰了，但仍然难以看清所有的点。解决此问题的一个方法是从不同的角度观看图，通过改变变量的顺序可以轻松实现这一点，如图 17-4 所示。更好的做法是，通过添加如下代码中的参数 `revolutions = n`，你可以让图形在屏幕上旋转 *n* 次，以便从各个角度进行观察。尝试一下吧！

```
# 图17-4
library(car)
library(rgl)
attach(trees)

scatter3d(Girth, Volume, Height, revolutions = 2)
rgl.snapshot("ch17.4.png", fmt = "png") # 存入工作目录
```

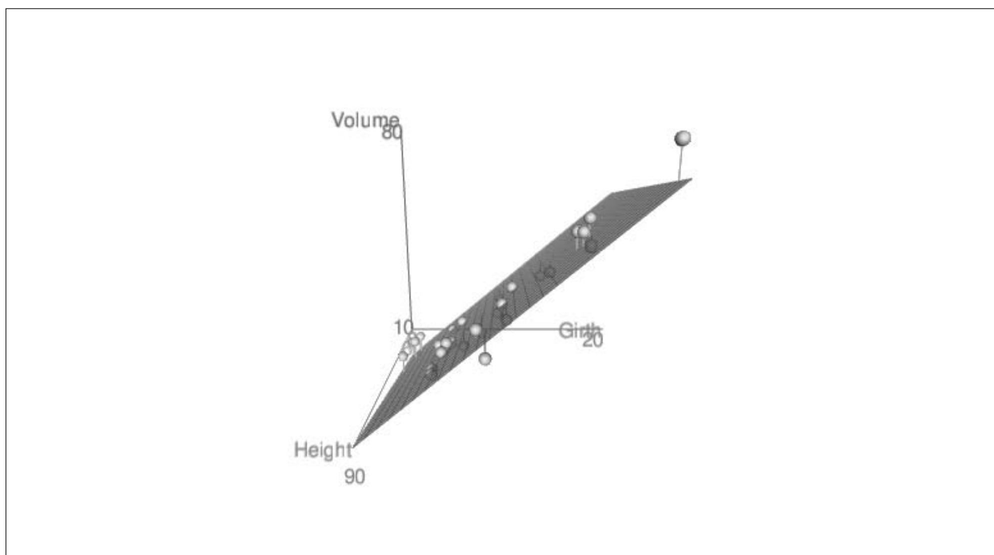


图 17-4：从另一个视角查看图 17-3。参数 `revolutions = 2` 可使其在屏幕上旋转 2 次（另见彩插）

`scatter3d()` 有许多自定义选项，如颜色、平面网格上的线、转速等。获取更多信息，请输入 **`?scatter3d`** 查询。

17.2 伪色图

三维散点图展现水平、垂直和对角线三个维度。这样做有时可行，有时却令人困惑。第三个维度还有一种展现方法，就是使用颜色渐变来给出深度感。这类图被称为伪色图（false-color plot），这种三维图是由 `lattice` 包中的 `levelplot()` 函数实现的。例子如图 17-5 所示，两个变量用空间表示，第三个维度通过颜色强度表示。此处绘制的 `coalash` 数据集来自 Gomez 和 Hazen（1970），可在 `sm` 包中找到。

纵轴和横轴代表偏北和偏东方向，而渐变颜色代表 `coalash` 的总量。从图上很容易看到浓度有多高以及相应的精确位置。生成该图的代码如下。

```
# 图17-5
library(lattice)
library(sm)
data(coalash)
attach(coalash)
levelplot(Percent ~ East*North)
```

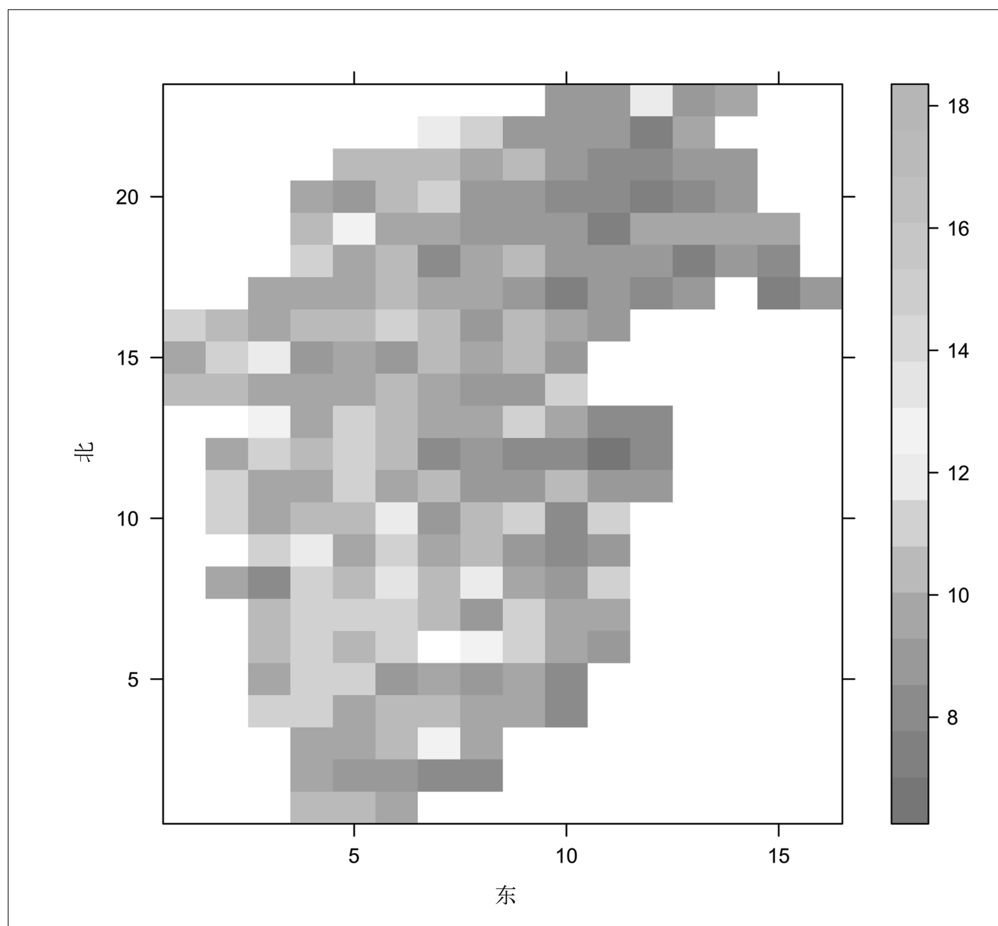


图 17-5：通过 lattice 包中的 levelplot() 生成的 coalash 数据的伪色图。任意点上 coalash 的总量由颜色梯度表示（另见彩插）

17.3 气泡图

另一种展现三维数据的图形是气泡图（bubble plot）。在这种类型的图中，两个变量绘制在 *x* 轴和 *y* 轴上，而第三个变量由图上的圆的面积或“气泡”表示。基础 R 中的 symbols() 函数可以创建气泡图，但我发现 PlotBubble() 更易于使用，你可以在 DescTools 包中找到它的语法如下：

```
PlotBubble(x = x-variable, y = y-variable,
  area = var represented by bubble,
  col = bubble color,
  border = color of bubble border,
  inches = diameter of largest bubble)
```

参数 x、y、area 和 col 都是必需的。请注意，因为圆的面积与半径的平方成比例，所以要使 area 变量与气泡所代表变量的平方根（square root）成比例。否则，气泡之间的大小差距可能会比较悬殊。另一种思考方式是，变量的大小应该由气泡的面积而非气泡的直径来表示。幸运的是，plotbubble() 会自动实现这种表示方法。如果使用 symbols()，就必须调整 Volume。我用两个函数分别运行了几个同样的问题并测量气泡，如果不这样做，我就无法打消顾虑。你可能也需要这样做。

首先，考虑 trees 数据，用气泡表示变量 Volume。图 17-6 的代码（如下所示）表明，Volume 已分配给参数 area。trees 数据的气泡图可能比基于相同数据的三维图更清楚一点。这是因为数据量小且参数 inches 设置适当，几乎没有重叠的气泡（尝试设置不同大小的 inches，看看结果如何）：

```
# 图17-6
library(DescTools)
attach(trees)
PlotBubble(x = Height, y = Girth, area = Volume,
  col="steelblue", border = "burlywood",
  inches = .25,
  xlab = "Height", ylab = "Girth",
  main = "Tree volume, proportional to circle area",
  family = "HersheySerif", font.main = 4,
  col.main = "maroon")
```

图 17-6 展示了该脚本创建的气泡图。

为了举例说明气泡图的另一个用途，我们考虑一个更大的数据集：来自 Sleuth3 包的 ex0923。数据取自对男性和女性收入的研究，包括了受访者的教育程度和智商。我们的目标是生成一个显示 Educ、AFQT 和 Income2005 三个定量变量的图，通过 Gender 进行划分。也就是说，图中会显示四个变量。在 PlotBubble() 中，参数 area 对 Income2005 进行操作，但是这个数据集存在一个问题，因为少数收入值非常高，超过了 50 万美元。PlotBubble() 显然调整过度了，在图上留下了太多空白，而且迫使所有点挤在一起。要修正这种情况，可以把所有的收入除以 1000，由此调整数据。这样可以使所有个人收入之间的关系保持不变，同时还解决了拥挤的问题，代码如下：

```
# 图17-7
library(DescTools)
library(Sleuth3)
attach(ex0923)
PlotBubble( x= Educ, y = AFQT, area = Income2005/1000,
  col = SetAlpha(as.numeric(Gender)), border = "burlywood",
  inches = .5, xlab = "Education", ylab = "AFQT test score")
title(main = "Income, proportional to circle area")
legend("left", c("Female","Male"),
  text.col = c(1:2), cex =.9, bty = "n")
```

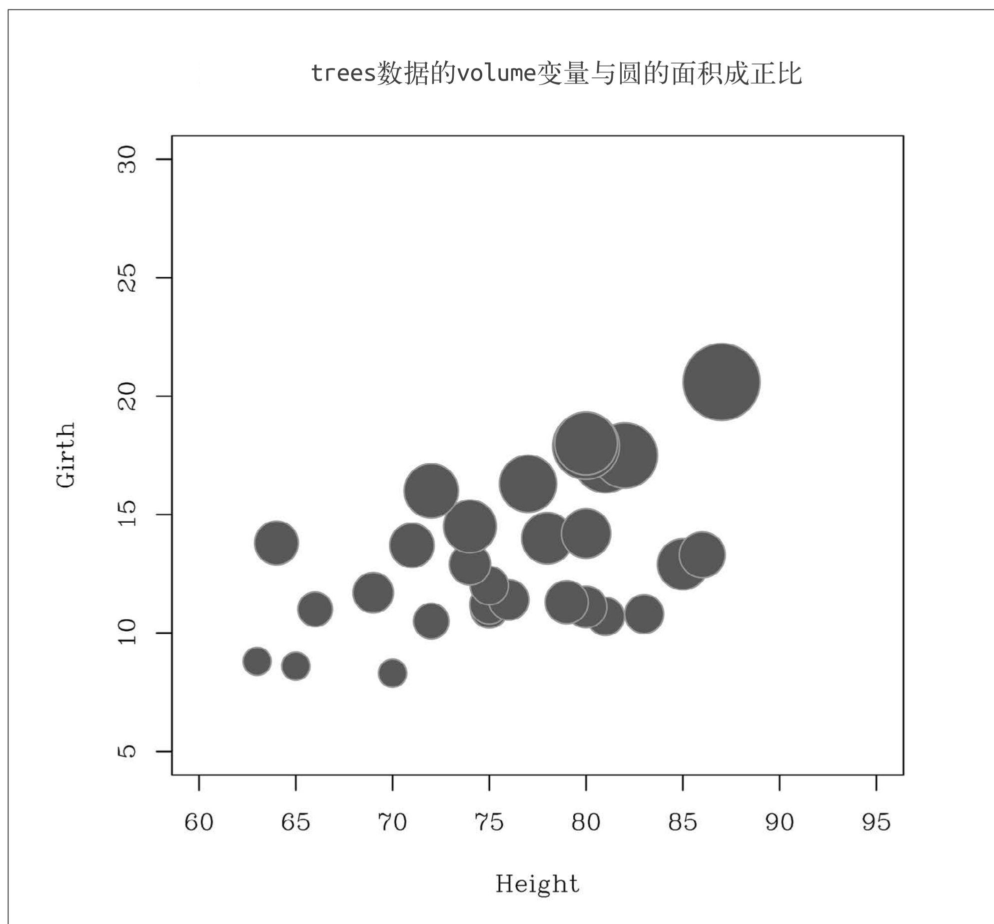


图 17-6：trees 数据的气泡图

参数 `col = SetAlpha(as.numeric(Gender))` 使 `Gender` 的两个值颜色不同。参数 `inches = .5` 使最大的气泡半英寸宽，并以其为参照，缩放其他所有气泡为合适的大小。在 `legend()` 命令中，`Gender` 的两个值是按字母顺序排列的，确保为颜色分配正确的名称。

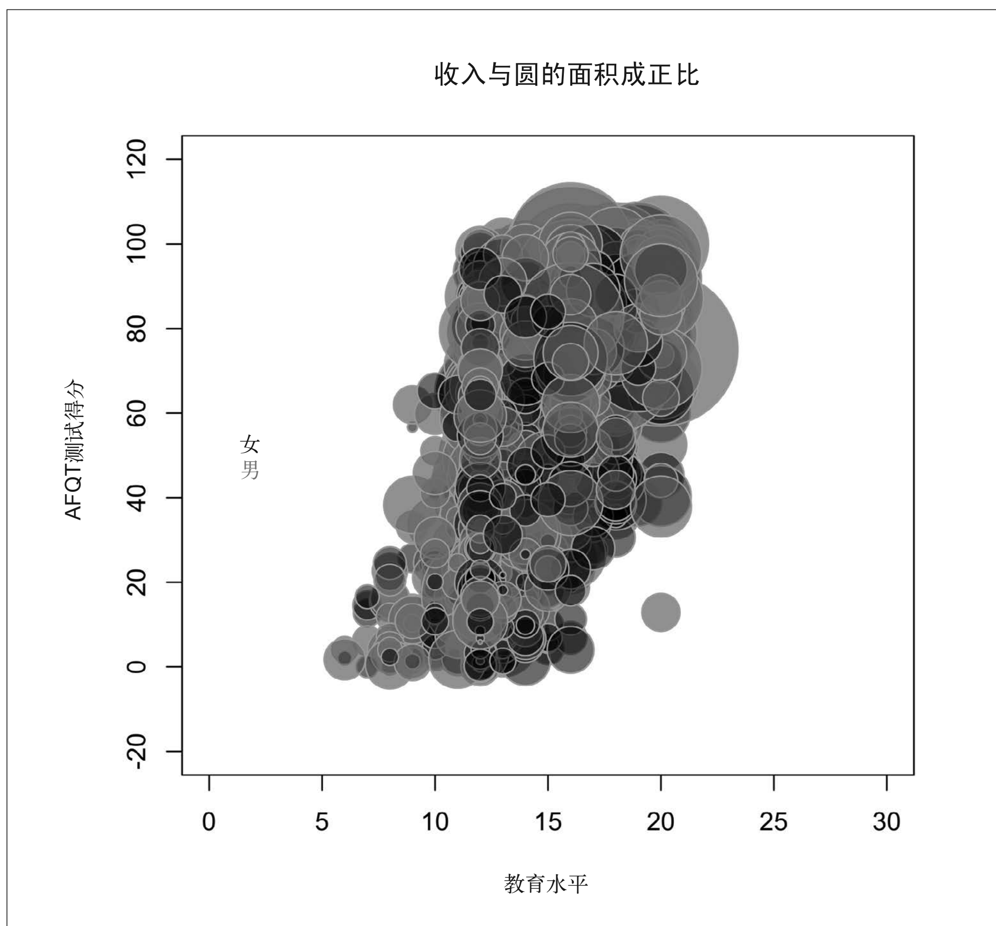


图 17-7：与教育水平和智商相关的收入的气泡图

气泡图显示，Educ 和 AFQT 是相关的，但我们很难就收入得出结论，因为有很多圆圈重叠在了一起。增大 inches 会雪上加霜，减小 inches 才是有用的举措。试着把 inches 设置为不同的尺寸，看看效果如何。你可能不会找到一个非常合适的值，因为对于大量数据来说，气泡图效果并不好。为了查看样本大小会产生什么样的影响，我们对数据集进行随机抽样，并绘制新的气泡图。我们把数据从 2500 多个观测值削减到 100 个，用 sample() 函数选择一个有 100 行数据的随机样本即可。此后，使用 [] 方法（查看 12.1 节回顾该方法）查看完整数据集的子集，只保留来自随机样本的行，但保留所有的列。值得注意的是，由此得出的结果子集是较大数据集的随机抽样，因为 samp = ex0923[s,] 中的行来自随机样本。完整的脚本如下：

```
# 图17-8
library(DescTools)
library(Sleuth3)
```

```
attach(ex0923)

# 取自ex0923的随机样本
set.seed(3)              # 每次均获取相同的随机样本
s = sample(nrow(ex0923), 100) # 随机抽样100行ID
samp = ex0923[s,]        # ex0923中和s中相同的所有行的所有列

detach(ex0923)           # R不会使用整个ex0923数据集
attach(samp)             # R将使用子数据集
PlotBubble( x= Educ, y = AFQT, area = Income2005/1000,
  col = SetAlpha(as.numeric(Gender) +3), border = "burlywood",
  inches = .25, xlab = "Education", ylab = "AFQT test score")
title(main = "Income, proportional to circle area")
legend("left", c("Female","Male"),
  text.col = c(1:2)+3, cex =.9, bty = "n")
```

结果如图 17-8 所示。

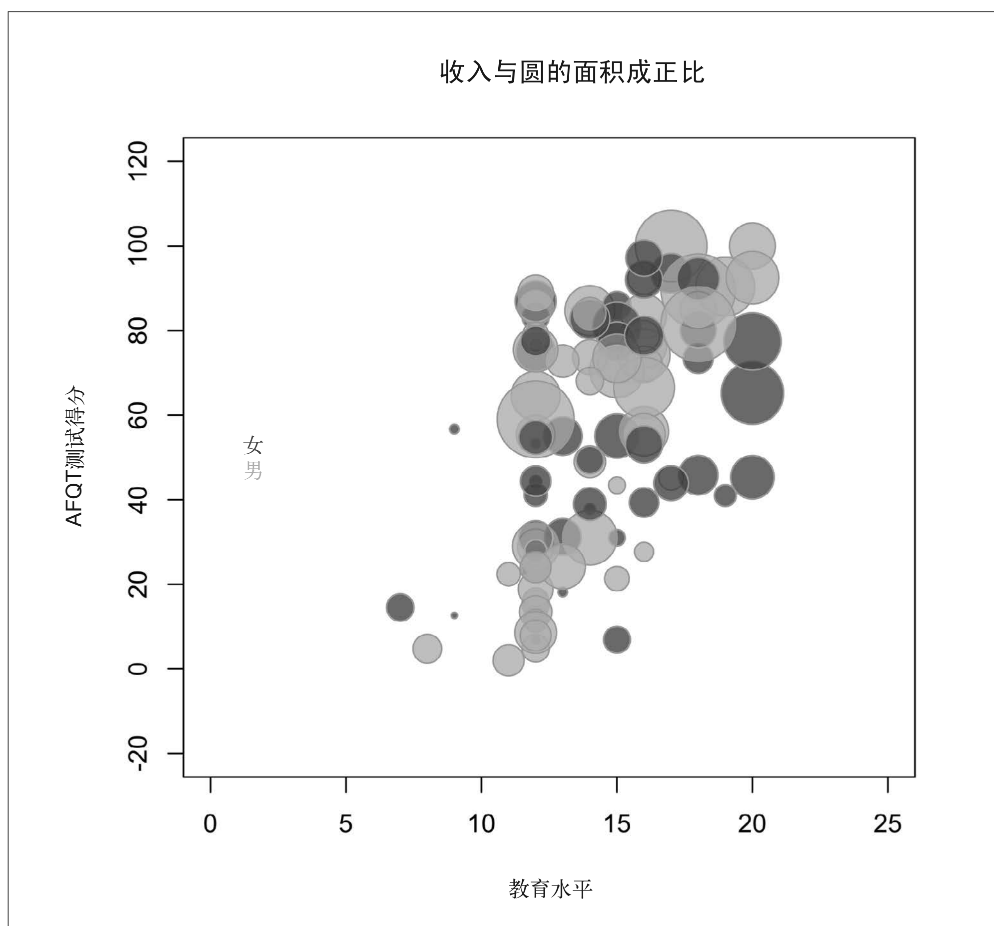


图 17-8：ex0923 数据集的随机抽样，关于教育水平、智商、收入和性别的气泡图

参数 col = SetAlpha(as.numeric(Gender) +3) 使 Gender 的两个值的颜色不同，并按色阶用三步设置颜色，实现命令如下：

```
> library(DescTools)
> PlotPar()
```

因此，图 17-6 使用的颜色为深蓝色（1 + 3）和浅蓝色（2 + 3），而没有使用默认的黑色（1）和红色（2）。注意，一定要用 +3 调整参数 text.col，使图例颜色和图上的圆圈匹配。这里还有其他颜色的调色板可以选择。查看其他颜色选项，请输入 **?hblue** 查询。

图 17-8 比非常密集的图 17-7 更易于阅读。我们可以清楚地看到，收入随受教育程度和智商的增长而增长。男性和女性的收入也容易比较了。在大多数情况下，当其他因素大致相等时，图中男性的气泡更大，意味着收入更高。

17.3.1　练习17-1

使用三维散点图研究 epicalc 包的 SO2 数据集中 deaths、smoke 和 SO2 的关系。在垂直轴上绘制 deaths，并解释它与其他变量的关系。

17.3.2　练习17-2

使用和上一道习题相同的数据，在伪色图中将 deaths 设为伪色变量。你从图中得出了什么结论？

第 18 章

协同图

协同图

有时，两个变量之间所呈现的明显的相关性可能是误导性的。这可能是因为一个或两个变量与第三个变量之间有强关联。以 car 包的 States 数据集为例。这是 SAT 考试的数据，在美国许多学生在考大学的过程中都会参加该测试。States 还包含其他几个关于 1992 年州级中学教育的变量。该数据集中有 51 个观测数据，代表各个州以及哥伦比亚特区。图 18-1 所示为 SAT 数学测验 SATM 的平均得分与各州公共教育花费总额（以千美元 / 学生为单位）的散点图。

生成图 18-1 的代码如下。

```
# 图18-1
library(car)
attach(States)
plot(dollars,SATM,
 pch = 16,
 col = "maroon")
grid(lty = "solid")
```

图 18-1 似乎表明教育花费相对较少的州的 SATM 得分较高，而支出较高的州的得分相对较低。这完全不合常理。我们预计（或至少期望）教育投入越多，则得分就越高。是不是有什么其他因素在影响结果呢？

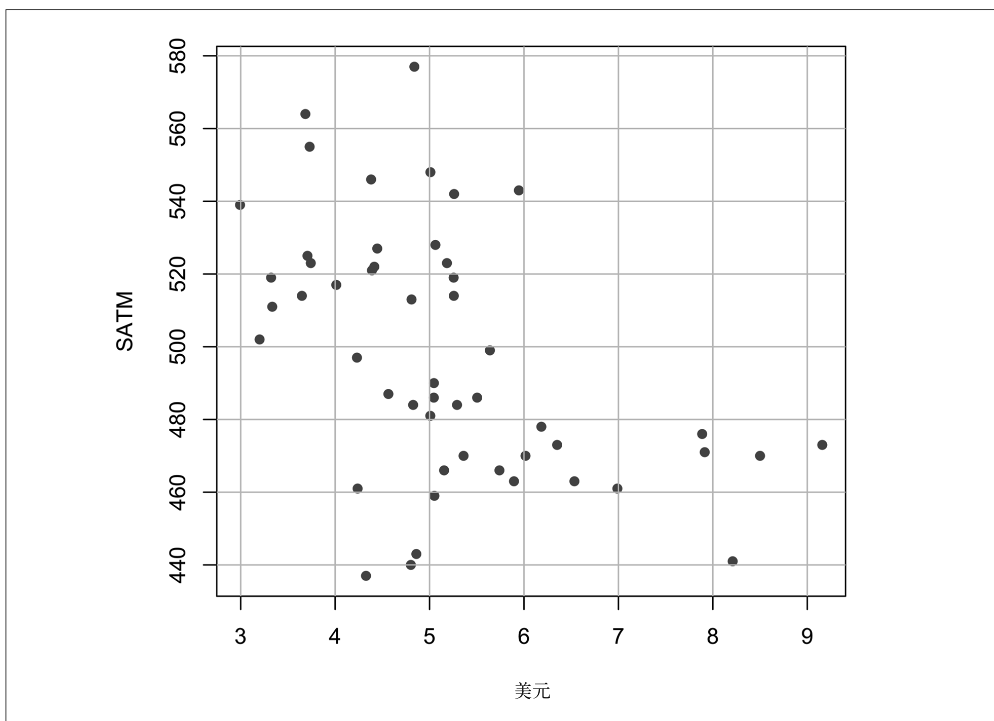


图 18-1：各州的 SATM 平均得分与公共教育花费总额（以千美元 / 学生为单位）的散点图。51 个点分别代表各州以及哥伦比亚特区

该数据集包括一个被称为 `percent` 的变量，它是参加 SAT 考试的应届毕业生的百分比。各州参加考试的学生的比例不同，测试平均分是否也有所不同？在参加考试的学生较少的州，是不是只有表现较好的学生参加了考试？也许在几乎每个人都参加考试的州，那些不太有天赋或是不太积极的学生拉低了平均分。我们可以用条件图 [conditioning plot，或协同图（coplot）] 来研究这个问题。这里的思路是把数据分片，以便查看在条件变量 `percent` 取不同的值时，`SATM` 和 `dollars` 的散点图。若所有散点图看起来都相同或非常相似，则表明 `percent` 不影响输出结果。若图看起来相当不同，则表明 `percent` 确实影响了 `SATM` 和 `dollars` 之间的关系。`coplot()` 函数用到的公式形式如下：

```
y ~ x | z
```

在这里，*y* 是垂直轴，*x* 是水平轴，*z* 是条件变量。以两个变量 *a* 和 *b* 为条件也可以，在这种情况下，公式为 *y* ~ *x* | *a* * *b*。生成图 18-2 中协同图的脚本如下：

```
# 图18-2
library(car)
attach(States)
coplot(SATM ~ dollars | percent,
 pch = 16,
```

```
    col = "royalblue",
    bar.bg = c(num = "goldenrod2"))
```

图 18-2 显示了 6 张散点图，每张图对应一个特定取值范围内的 percent 数据“片”或子集。图顶部的框中显示了一组 6 个条，每个条表明每张散点图所涵盖的百分比范围。左下角的条表明左下角的图涵盖了 percent 不高于 12% 的州。从下往上数的第二条表示下面一行中的第二张图涵盖了 percent 约为 8%~16% 的州。最上面一条表示右上角的图涵盖了 percent 不小于 54% 的州。该协同图似乎和之前提出的假设一致：参加考试的学生比例较低的州得分最高，参加考试的学生比例较高的州得分较低。此外，这 6 张图中似乎都看不出 SATM 和 dollars 之间有任何显著关联。

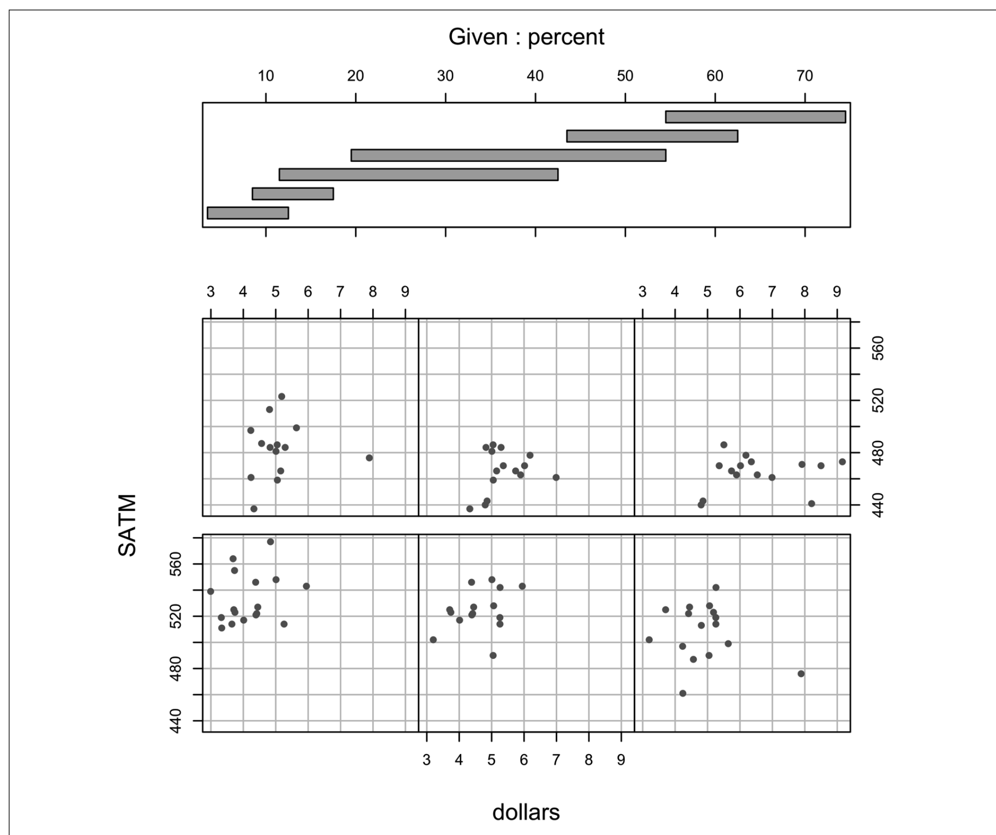


图 18-2：SATM 和 dollars（美元）的协同图，以 percent（参加 SAT 考试的应届毕业生的百分比）为条件变量

图 18-2 中的条存在重叠，这意味着一些州的情况在 2 张或 3 张图中都有所体现。R 默认这样做，以确保每一张图中有足够多的点，从而能绘制出一张有用的图。注意，每一张图中大约有 15~17 个点。如果每张图都不重叠，那么每张图仅有 8 或 9 个点（51 除以 6），根

据分隔点的位置，点也可能更多或更少。在这种情况下，R 默认的选择似乎已经完成了我们所期望的，但情况并不总是这样。

使用参数 number 可以控制分片的数量，使用参数 overlap 可以控制单个片重叠的程度，如下例所示：

```
# 图18-3
library(car)
attach(States)
coplot(SATM ~ dollars | percent,
  pch = 16,
  col = "royalblue",
  bar.bg = c(num = "seagreen"),
  overlap = 0,
  number = 5)
```

从图 18-3 中可以看到，现在只有 5 片，且互不重叠。注意，在所有图中，R 选择的切割点的位置，会以所有图中点的数量大约相等的方式来创建分片。

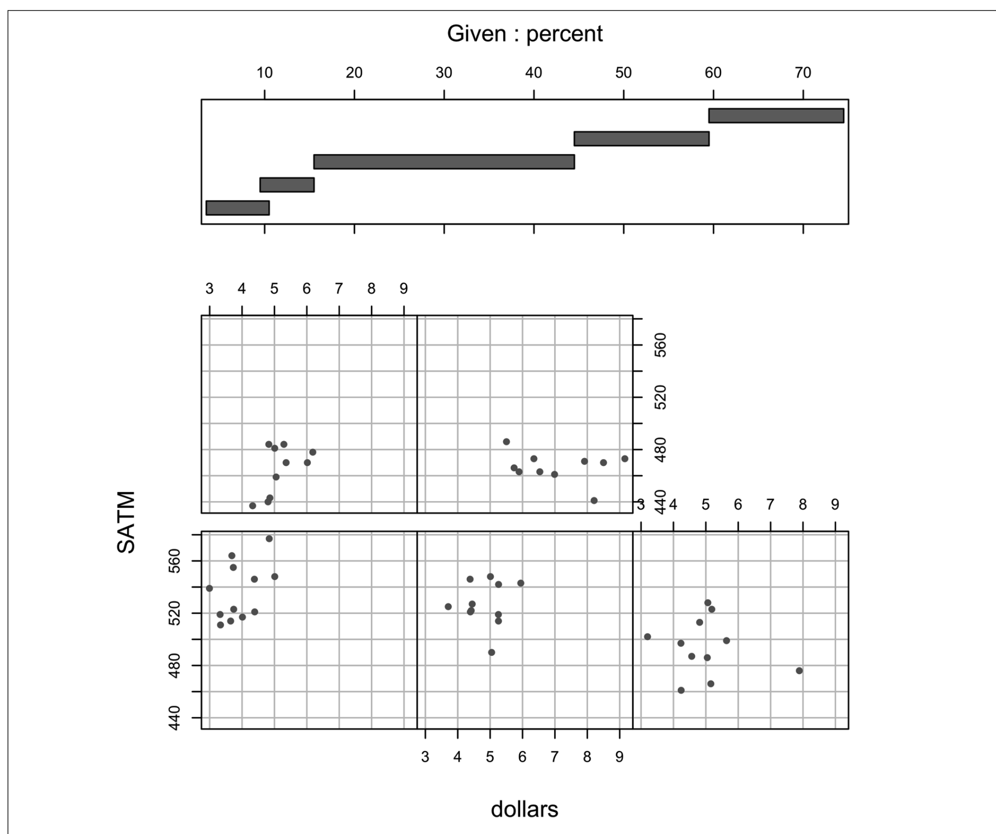


图 18-3：以 percent 为条件变量的 SATM 和 dollars 的协同图，分为 5 片，各点无重叠

即使我们放弃控制切割点，而交由 R 处理，图 18-3 中的结果图看起来仍然很好。在某些情况下，我们可能需要找到精确的切割点，而不去利用 R 的智能。这是可以做到的，但需要花费精力。假设我们想要 4 张不重叠的图，并选择精确的切割点。我们需要创建一个 4 行矩阵，每行对应一张图。每行将有两个数字：图中最低的 `percent` 和最高的 `percent`。矩阵名将提供给参数 `given.values`。矩阵如下所示：

```
 0 19.9
20 39.9
40 59.9
60 75
```



获取更多关于创建矩阵的信息，请输入 **`?matrix`**。

图 18-4 中 4 片的 `percent` 宽度完全相等（最高的片除外），但各图中的点数完全不同，其脚本如下：

```
# 图18-4
library(car)
attach(States)
mat = matrix(c(0,19.9,20,39.9,40,59.9,60,75),
 byrow = T,
 nrow = 4,
 ncol = 2)
coplot(SATM ~ dollars |percent,
 pch = 16,
 col = "royalblue",
 bar.bg = c(num = "maroon"),
 given.values = mat)
```

结果如图 18-4 所示。

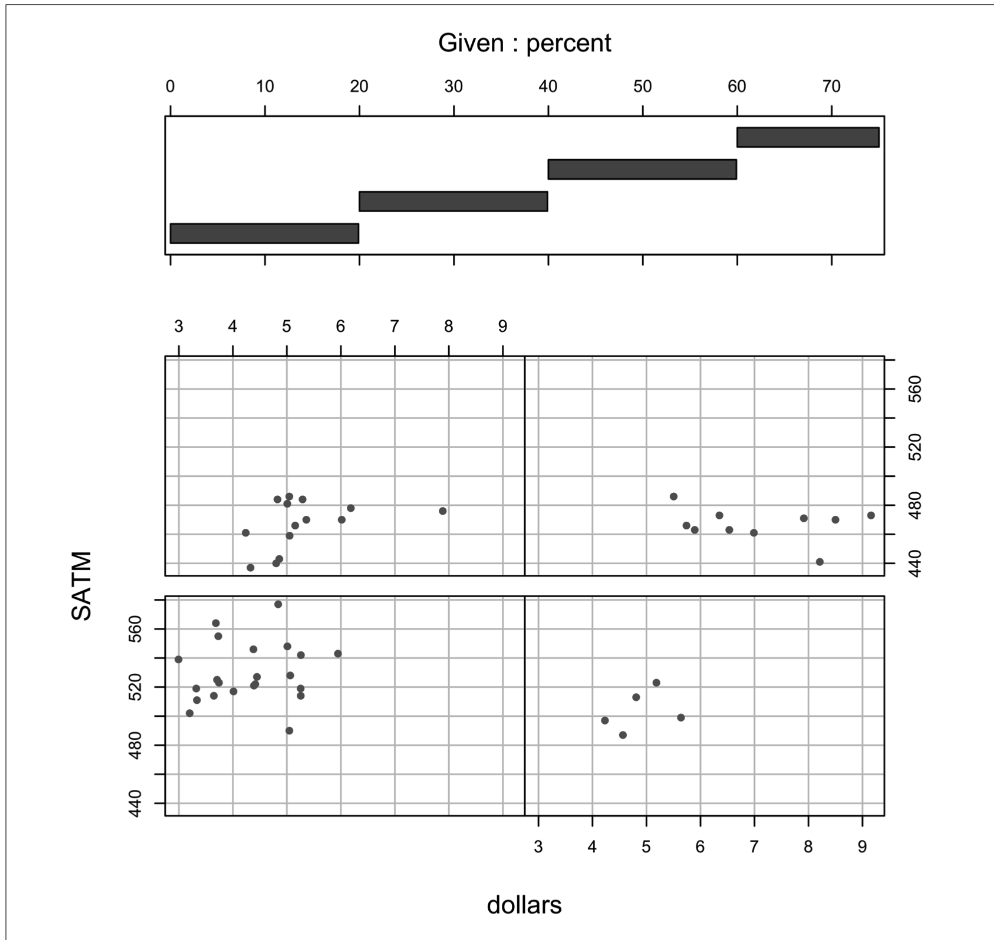


图 18-4：以 percent 为条件变量的 SATM 和 dollars 的协同图，分为 4 片，自定义精确切割点

percent 和 SATM 之间依然呈现出很强的联系，我们的结论并没有改变。在这种情况下，一个面板中的点不少于 5 个。然而，一个或多个面板包含 0、1 或 2 个点的情况也可能出现，这样的面板没有什么价值，或难以解释。因此，如图 18-2 所示，R 默认重叠各条，以避免面板上的点过少甚至为空。我们总是可以选择创建一个面板不重叠的协同图。当给定足够多的点时，图形往往更易于解释。

练习18-1

污染会造成死亡吗？研究死亡率与大气污染的关系，数据来自 sleuth2 包的 ex1123 数据集。二者之间的明显关联是否可以用其他因素来解释？

第 19 章

聚类分析：树状图和热图

19.1 聚类分析

聚类分析（clustering）是指一系列探索多变量数据的相关方法。R 中有几十个可用的聚类分析函数。本章我们只关注其中一个：基础 R 中的 hclust() 函数。该函数可以进行*层次聚类分析*（hierarchical clustering），这是最常用的聚类技术之一，总体来说非常适合入门级的聚类分析。它的思路是把观测值放到集群或分组中，其中单一集群中的成员彼此相似，且和其他集群中的观测值不同。此外，某个集群可能被判断为与其他集群有不同程度的相似之处。我们将通过*树状图*（dendrogram）来理解集群间的关系，它看起来像一棵倒置的树。本章后面的图 19-2 呈现了一个树状图的例子。

考虑 mtcars 数据集，它来自《汽车族》杂志 1974 年关于多款新车型特点的报告。我们用 head() 函数来看一下该数据集的前 6 行数据，代码如下：

```
> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0
                  gear carb
Mazda RX4            4    4
Mazda RX4 Wag        4    4
Datsun 710           4    1
Hornet 4 Drive       3    1
Hornet Sportabout    3    2
Valiant              3    1
```

我们想把各种各样的车型分为集群，把相似的汽车放在同一集群中。有两种办法可以做到这一点。一种是聚合方法（agglomerative method），首先创建一个包含最匹配对的集群，然后创建一个匹配度次之的集群，其中的对数据可以是单个观测值对，或者是单个观测值和已存在的集群对，以此类推，直到所有观测值都在一个大集群中。另一种是分裂方法（divisive method），即将总集分为子集，子集再分为更小的子集，以此类推。虽然有几种方法可用，但 `hclust()` 函数使用聚合方法。我们将使用默认的“complete”方法。

我们应该如何测量两个观测值间的相似性或者距离呢？这需要找到一个测量办法，融合所有可用的信息，以确定一种车型和另一种车型之间的“距离”。如果只考虑一个变量，那么我们显然应该选择测量车型在这个变量值上的绝对差异。然而，在我们的例子中有 11 个变量，所以我们希望距离测量考虑到所有变量。我们从一个较简单的例子开始。假设有两辆车 Car-1 和 Car-2，每辆车都对两个变量 x 和 y 进行了测量。因此，Car-1 对应点 (x_1, y_1)，Car-2 对应点 (x_2, y_2) 。这两点展示在图 19-1 左上方的图中。图 19-1 右上方的图以实线标出了两点之间的最短距离。

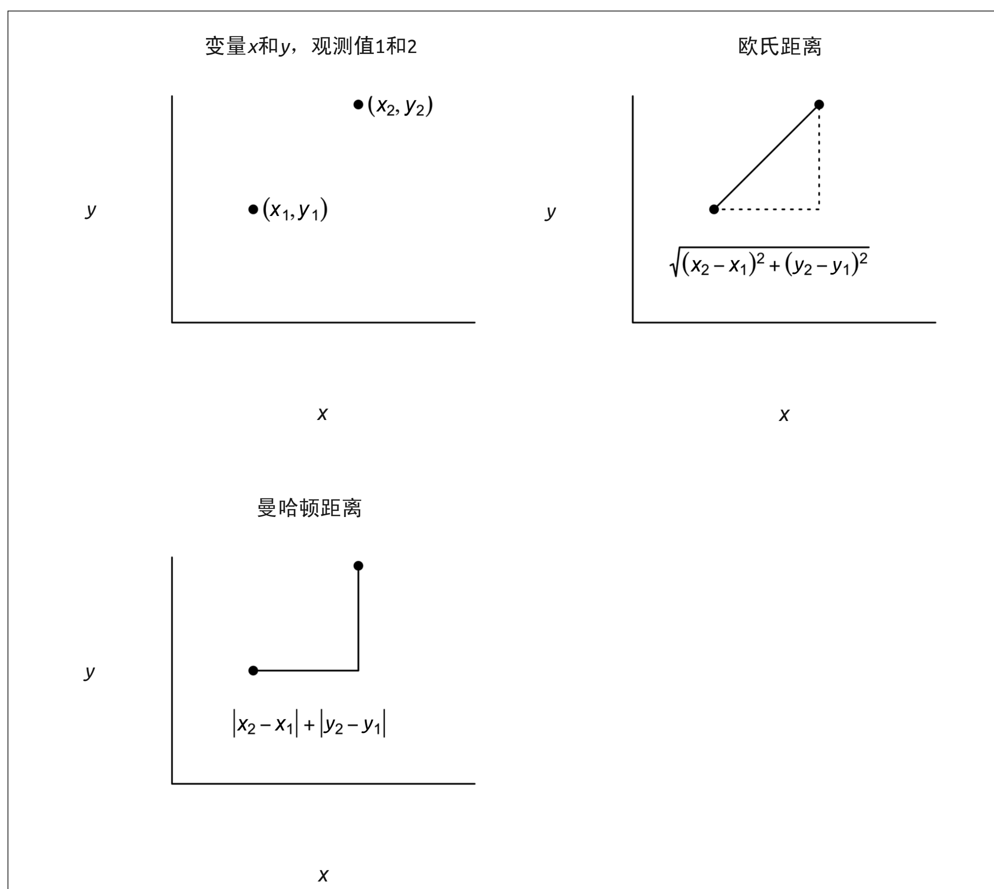


图 19-1：二维空间中的距离测量

注意，这条线是直角三角形的斜边，因此很容易计算距离，如下所示：

```
distance = sqrt[(x2 - x1)^2 + (y2 - y1)^2]
```

记住，斜边平方等于两边的平方和。该距离被称为欧氏距离（Euclidean distance），是 R 中的 dist() 函数默认使用的距离测量方法。此外，还有其他几种测量距离的方法。图 19-1 的左下图展示了其中一个称为曼哈顿距离（Manhattan distance）的方法，也叫“出租车几何”或“城市街区距离”。根据你所解决的具体问题，该方法或许更合适。R 提供了这种方法和其他几种可用的方法，但我们将继续用欧氏距离处理该问题。如果有 3 个变量，你可以按如下方式扩展欧氏方法：

```
Euclidean distance = sqrt[(x2 - x1)^2 + (y2 - y1)^2 +
  (z2 - z1)^2]
```

同理，可以将测量扩展，尽可能涵盖更多变量。

在图中加入数学表达式

有时，一个数学公式或表达式就可以大大提升图形效果。幸运的是，R 允许将 expression 作为 text()、mtext()、axis() 和 legend() 中任意一个函数的参数来添加这样的表达式。下面的脚本以 text() 命令包含数学表达式，生成了图 19-1：

```
# 图19-1的脚本
par(mfrow = c(2,2))
x = c(2,5)
y = c(3,6)
yp = c(0,6)
xp = c(0,8)

plot(x,y, pch = 16, xlim = xp, ylim = yp,
  xaxt ="n", yaxt = "n", bty ="l",
  main="Variables x and y, Observations 1 and 2",
  cex.main = .9,
  ylab = "")
text(x = 3.2, y = 3,
  labels = expression(group("(", list(x[1], y[1]), ")")))
text(x = 6.2, y = 6,
  labels = expression(group("(", list(x[2], y[2]), ")")))
mtext(text = "y",
  side = 2, las = 1,
  cex = .8, line = 3)
plot(x, y, pch = 16, type = "o", xlim = xp, ylim = yp,
  main = "Euclidean distance",
  xaxt = "n", yaxt = "n", bty = "l", ylab = "")
text(3.6, 1.5, labels =
  expression(sqrt((x[2] - x[1])^2 + (y[2] - y[1])^2)))
lines(x, y, type = "s", lty = "dotted")
mtext(text = "y", side = 2, las = 1, cex = .8, line = 3)
```

```
plot(x,y,
  pch = 16, xlim = xp, ylim = yp,
  main = "Manhattan distance",
  xaxt = "n", yaxt = "n", bty = "l", ylab = "")
lines(x,y,type="s" )
text(3.6, 1.5,
  labels = expression(group("|", x[2] - x[1],"|") +
  group("|",y[2] - y [1],"|")))
mtext(text = "y", side = 2, las = 1,cex = .8, line = 3)
```

关于使用细节，请参考 plotmath 帮助文件。

mtcars 中变量的值差别很大。例如，disp 的值远远超过 100，但 cyl 的值均为个位数。这意味着，在确定距离方面，如果仅考虑测量值的范围，disp 对距离的影响远胜 cyl。想象一下，两个测量长度的变量，一个以英寸为单位，而另一个以英尺为单位，那么完全相同的距离将会记录为完全不同的数字，较大的数字对欧氏距离的影响较大。因此，一定要将所有变量转换为可比较的测量尺度。

我们可以运用一个简单的变换来进行*规范化*（normalize）或*标准化*（standardize），也就是使每个变量的平均值为 0，标准差为 1。我们先在 mpg 上尝试使用这种变换。首先获得 mpg 的均值和标准差，代码如下：

```
> mean(mpg)
[1] 20.09062
> sd(mpg)
[1] 6.026948
```

从 mpg 的每个值中减去均值并用结果除以标准差，将获得一个均值为 0，标准差为 1 的变量 mpg，代码如下：

```
> mpg2 = (mpg - 20.09)/6.026948
> mean(mpg2)
[1] 0.0001037009        # 小舍入错误!
> sd(mpg2)
[1] 1
```

这种规范化非常常见，因此 R 为这个过程提供了一个函数，使其只需一步操作，代码如下：

```
> mpg3 = scale(mpg)
> mean(mpg3)
[1] 7.112366e-17   # 很小,很小;实际上 = 0
> sd(mpg3)
[1] 1
```

幸运的是，我们不需要按比例缩放每个变量，而可以一次操作整个矩阵。现在我们将数据框转换为矩阵，对按比例缩放的矩阵进行距离测量，计算集群，并绘制树状图。代码如下：

```
# 图19-2
attach(mtcars)
cars = as.matrix(mtcars)  # 转换为矩阵,dist需要它
h = dist(scale(cars))     # 缩放cars矩阵并计算dist矩阵
h2 = hclust(h)            # 聚类分析
plot(h2)                  # 绘制树状图
```

图 19-2 中的树状图显示了聚类分析的结果。

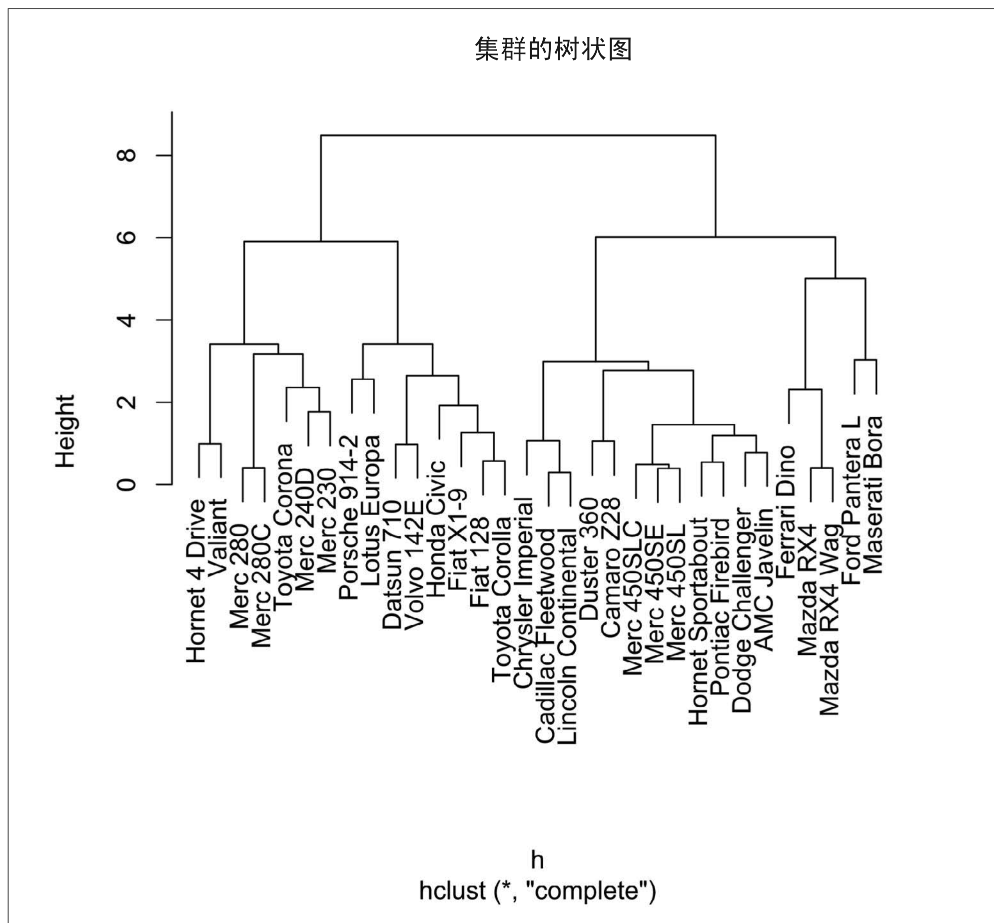


图 19-2：mtcar 数据集中集群的树状图

垂直标尺为“高度”（Height），可以帮助我们理解集群的形成过程。这个图形看起来就像很多 U 形钉把同一集群中的观测值连接了起来。沿 Height 标尺观察，U 形钉水平部分越向下，相关集群形成得越早。因此，高度接近零的 U 形钉是第一个形成的，因此最接近欧氏距离。

高度接近 8 的集群是最后形成的，因此相对较远。彼此相邻的集群的距离不一定短！例如，图右侧两个 Mazda 车型非常接近，在高度约 1 的位置已经形成了一个集群。而 Ford

Pantera 紧挨着 Mazda 的集群，但和 Mazda 不是特别接近，因为它们直到高度达到 5 时才进入同一个集群。

通过转置，cars 矩阵也可以聚类变量，而不是观测值。具体做法就是使第一行成为第一列，第一列成第一行，以此类推，代码如下：

```
newcars = t(cars)  # newcars是cars的转置矩阵
h = dist(scale(newcars))
h2 = hclust(h)
plot(h2)
```

19.2 热图

查看热图（heat map）是获取 mtcars 数据集中所有数据概况的另一种方式。在此类可视化图中，标准化矩阵中的每个数字都被转换成一个颜色矩形。这个过程以系统的方式实现，以颜色表示数字的近似值或强度。例如，我们或许会用深红色代表非常小的数字，浅红色、橙色、黄色对应的数字不断增大，最后是代表最大数字的白色。这是 image() 函数默认的色域，除此之外还有许多其他颜色集可用。图 19-3 显示了 mtcars 数据集中按比例缩放值的简单热图，生成代码如下：

```
# 图19-3
attach(mtcars)
cars = as.matrix(mtcars)
image(scale(cars))    # 简单的热图
```

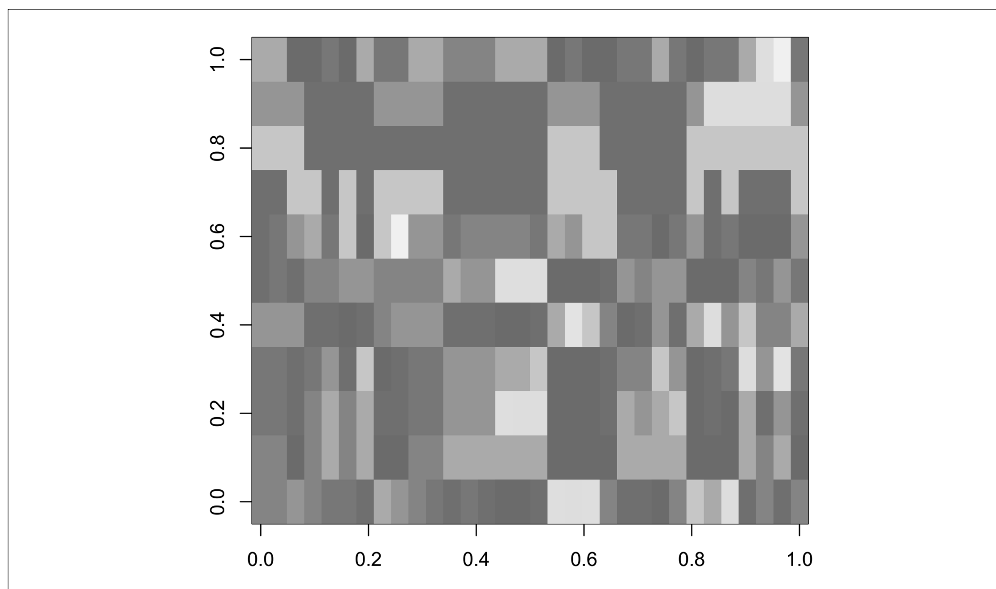


图 19-3：以默认颜色表示的 mtcars 数据集的热图（另见彩插）

参数 `col = rainbow()` 控制 `image()` 函数的色域。另一个可用的颜色方案是一系列蓝色（由深到浅）。下面的命令展示了如何调用一系列蓝色：

```
# 图19-4
image(scale(cars), col = rainbow(256, start = .5, end = .6))
# 蓝色调的热图
```

结果如图 19-4 所示。

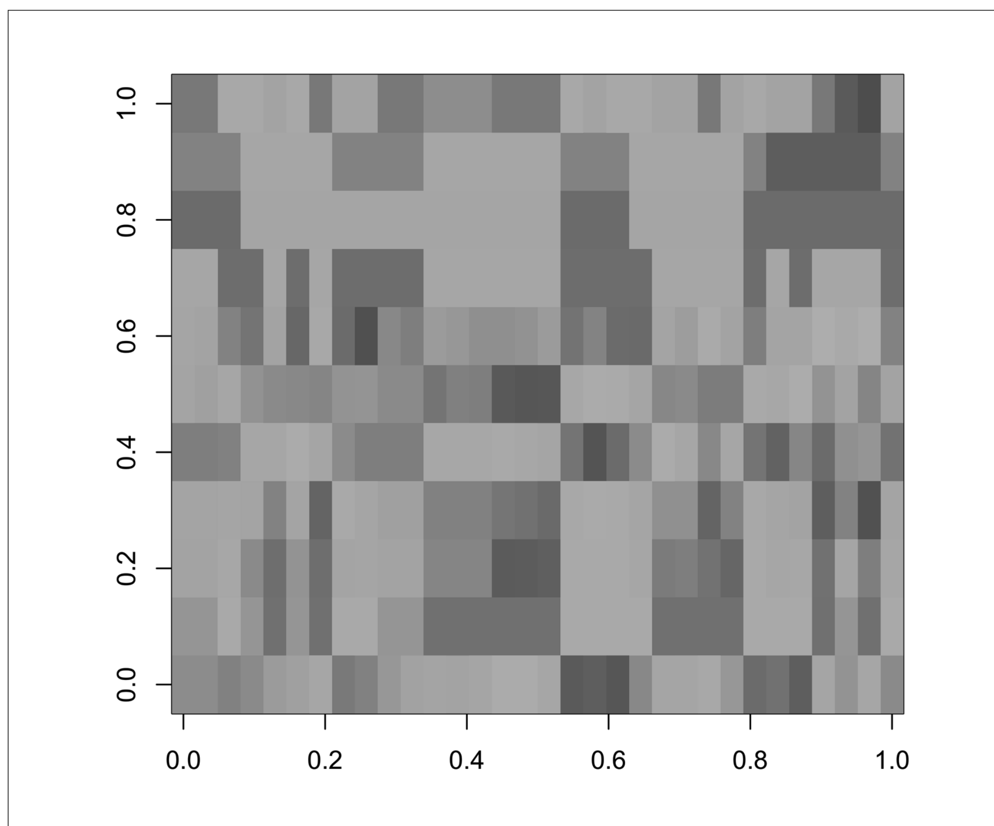


图 19-4：用一系列蓝色表示的 `mtcars` 数据集的热图（另见彩插）

不是所有的颜色集都很容易理解。什么颜色都有的图形让人困惑，例如，你很难知道深绿色和深蓝色哪个代表更大的数字。而对于大多数人来说，图 19-3 和图 19-4 中的配色方案是比较容易掌握的。参数 `rainbow` 中每一个 `start` 和 `end` 的值必须大于等于 0，但不大于 1，且两者的值不能相等。你可以尝试不同的值，看看能否找到一组值，和本例展示的两张图效果一样好。获取更多信息，请输入 **`?rainbow`** 查询。

图 19-3 和图 19-4 中的热图转向一边，如同数据矩阵倒向了左边。如果数一下，你可以发现图中有 11 行 32 列，而原始数据集为 11 列 32 行。虽然颜色显示了大范围的值，包括许

多深红色（低）值以及一些浅黄色和白色（高）的值，但图中似乎没有任何明显的模式。

正如在聚类分析中所做的一样，我们希望在数据中找出模式。实际上，我们可以将树状图和热图融合到一个可视化展示中，以帮助理解变量和特定车型之间的关系。`heatmap()` 函数可以在进行聚类分析的同时绘制热图。在这里，行和列会重新排序，把相似的放在一起，且单元格会涂成适当颜色。如下命令使用默认选项生成图 19-5。

```
> heatmap(scale(cars))      # 图19-5
```



关于许多可用选项的更多信息，请查看帮助文件。你可以查到如何包含一个行或列的树状图、用于测量距离的方法、如何加权行和列，等等。

图 19-5：mtcars 中集群的热图（另见彩插）

在图 19-5 中可以看到一些醒目的模式。注意颜色如何将一些车型组和其他组分开。将这些集群与左边的树状图进行比较，你可以看到，某些车型在相同的集群中，而且同一集群中的模型，其变量之间有相似的颜色模式，这一点在最早形成的集群中最为明显。

图 19-6 中显示了一张类似的热图。

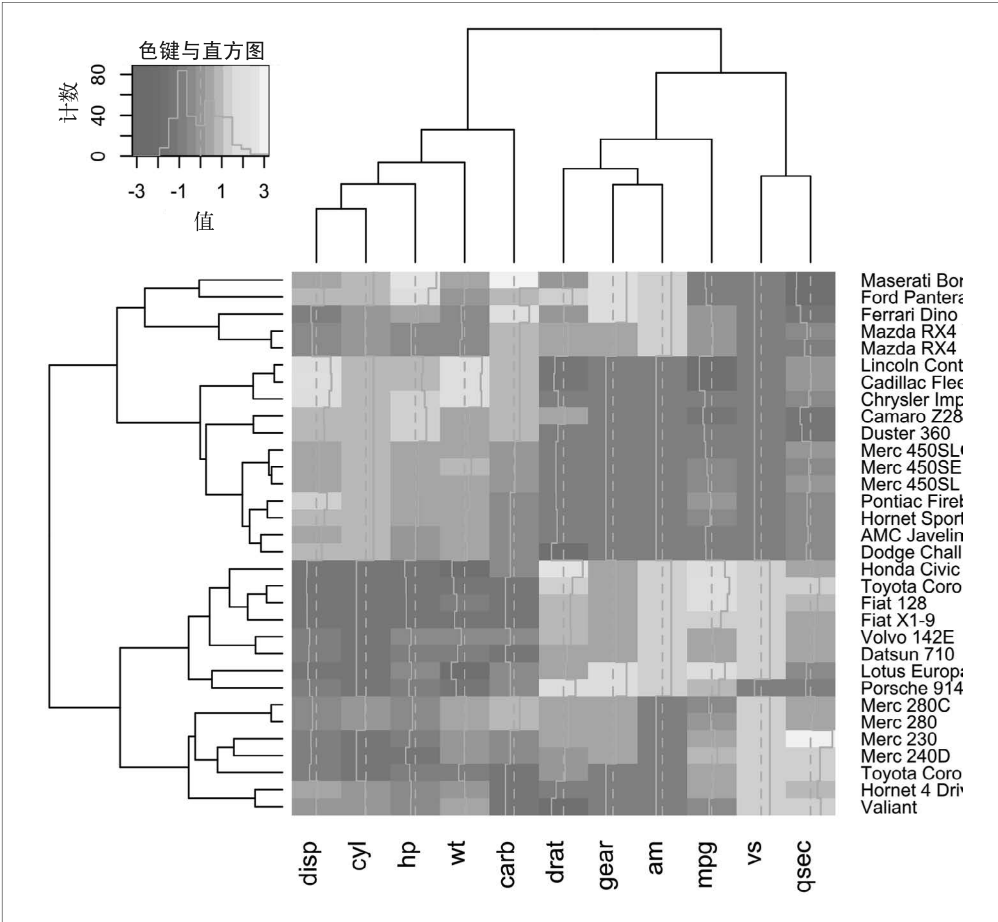


图 19-6：使用 gplots 包中的 heatmap.2() 生成的 mtcars 的热图（另见彩插）

图 19-6 是使用 gplots 包中的 heatmap.2() 函数生成的。heatmap.2() 提供了两个额外的特征，使图更易于理解。首先，左上角有一个键，表明了颜色与变量值之间的关系。其次，有一组垂直线贯穿每一列，其中虚线代表 0，实线表示特定单元中值与 0 的差异。这样加强了键的效果，方便确认每个单元。生成该图的代码如下。

```
# 图19-6
library(gplots)
heatmap.2(scale(cars))
```

聚类分析不是精确的科学分析，而是一种在复杂的数据中寻找规律的方法。聚类算法、树状图和热图是寻找规律时使用的工具。和其他的工具一样，它们可以帮助我们达成目标，如果我们使用时不小心，它们也会造成麻烦。上述讨论远非对聚类分析的完整的解释，而是一个引子，也许这可以激励你前进，继续学习。R 中还提供了许多其他的聚集分析和热图函数。

19.2.1　练习19-1

使用不同的聚类方法，为 `mtcars` 数据创建新的树状图。你可以用帮助函数（**`?hclust`**）查看有哪些可用的方法。不同方法创建的图形有什么不同？其他方法不一定会给出相同的答案。你可能会发现一个方法能很好地解决一个问题，却不能很好地解决另一个问题。此外，你还可以尝试以不同的方法测量距离。请输入 **`?dist`** 查看可用的方法。

19.2.2　练习19-2

分别使用下列颜色方案中的每一种，为 `mtcars` 数据绘制热图。

```
heat.colors
cm.colors
terrain.colors
topo.colors
rainbow.colors
```

某些图是否比其他图更易于阅读？你认为你会继续使用哪一个方案？有没有不打算使用的方案？

19.2.3　练习19-3

回想一下，我们在第 1 章中研究过的 `airquality` 数据有很多缺失值。`Amelia` 包中的 `missmap()` 函数使用一种简单的热图寻找缺失值。请安装并加载 `Amelia`，并找到 `airquality` 的缺失值。这张热图为什么比本章中讨论的热图更简单？该图形是如何帮助你理解数据集的？将其与 `epade` 包中 `missiogram()` 函数的输出进行比较。

第20章

马赛克图

绘制分类数据图形

目前为止，我们所研究的图大部分是关于定量变量的。在一些情况下，我们将定量变量和分类变量混合，通常使用分类变量的不同值来定义组，每一个组都有自己的图形。然而，有时我们关注的所有变量都是分类变量，这时就需要特殊的图形化方法。

我们来探讨 epicalc[1] 包中的一个数据集。你需要安装该包以及 vcd，其中包括一些处理分类变量的函数。相关命令如下：

```
> install.packages("epicalc")
> install.packages("vcd")
> library(epicalc)
> library(vcd)
```

我们将查看 ANCdata 数据集。你需要获得一些有关该数据集的信息，输入以下命令查询：

```
> ?ANCdata
```

该数据来自两家诊所，相关研究关注的是高风险孕妇的护理类型。这里有三个变量，均为分类变量，每个变量只有两个值（水平）。我们想知道围产期死亡率（即七天内新生儿夭折和死胎的概率）是否与治疗的类型以及提供护理的诊所有关。先来看一下 death（死亡率）与 anc（治疗频率）之间的关系。如下脚本中的 table() 命令将统计两个变量的组合

注 1：根据原书作者的补充说明，CRAN 上已经不再提供 epicalc 包，本章中出现的 epicalc 包均可替换为 epiDisplay 包。——编者注

中观测值的数目：

```
# 表20-1
library(epicalc)
library(vcd)
attach(ANCdata)
xtab1 = table(death,anc)   # 将此表作为对象 "xtab1"
xtab1    # 打印xtab1中的值
```

表 20-1：准妈妈的治疗频率和婴儿死亡率

```
     anc
death old new
  no  373 316
  yes  46  20
```

由此可知，接受旧式（old）治疗的孕妇生产后，有 373 名婴儿存活，46 名死亡。接受新式（new）治疗的孕妇中，有 20 人失去了孩子，316 人幸免于难。这张表很简短，但是需要花点时间，动动脑子才能处理好。哪种治疗效果更好？对概括的数据进行可视化的展示，答案更为明显。我们将使用马赛克图（mosaic plot），用长方形面积的大小表示单元格中的数目（每格的计数值）。注意，下面的 `mosaic()` 命令从对象 `xtab1` 中的表获取信息：

```
> mosaic(xtab1) # 用命令操作表,和原来的不同
```

另外，`mosaic()` 命令可以接受像公式一样的参数。因此，以下两个命令是等价的：

```
> mosaic(xtab1)
> mosaic(~death+anc)
```

生成图 20-1 的所有命令如下所示：

```
# 图20-1
library(epicalc)
library(vcd)
attach(ANCdata)
xtab1 = table(death,anc)
mosaic(xtab1)  # 或者 mosaic(~death+anc)
```

如果我只有一张频率表怎么办

科学杂志、报道或简报上的文章可能会给出频率表，这种表类似于我们为 ANCdata 数据集生成的表（见表 20-1）。也许你想继续进行分析，但无法获得原始数据。不要灰心，你可以创建一个简短的脚本，包含频率数据。这样就可以生成包括本章所有图在内的很多图。这种方式涉及创建一种类型的数据结构。我们还没有讨论过这个数据结构，它就是**数组**（array）。

`array()` 命令的基本形式如下：

```
array(data, dim = length(data), dimnames = list())
```

此处，*data* 是一个向量，*dim* 是表的维度向量，*dimnames* 是输出变量的名称。使用如下命令可以创建表 20-1：

```
xtab1 = array(c(373, 46, 316, 20), c(2,2),
  list(c("no", "yes"), c("old", "new")))
```

对象 xtab1 本质上和 table() 生成的 xtab1 是一样的。你也可以在脚本中使用下面的等效命令，把较长的 array() 命令分成较小的片段（一些人觉得这样更易于阅读）：

```
# 输入ANC频率表,而不是读取ANC数据
# 双向表,数据来自表20-1
library(vcd)
counts = c(373, 46, 316, 20)  # 输入第一列和第二列
death = c("no", "yes")
anc = c("old", "new")
xtab1 = array(counts, c(2,2), list(death, anc))
names(dimnames(xtab1))  = c("death", "anc")
xtab1           # 打印表20-1
mosaic(xtab1)   # 生成图20-1

# 三向表,数据来自表20-2
library(vcd)
cnts = c(176, 197, 12, 34, 293, 23, 16, 4)
clinic = c("A", "B")
death = c("no", "yes")
anc = c("old", "new")
xtab2 = array(cnts, c(2,2,2), list(clinic, death, anc))
names(dimnames(xtab2)) = c("clinic", "death", "anc")
xtab2          # 表20-2
mosaic(xtab2)  # 图20-2
# 其他图中使用合适的mosaic()命令
```

如图 20-1 所示，接受旧式疗法的孕妇略多，因为旧式疗法对应的块比新式疗法的大。婴儿死亡率在接受旧式疗法的人中较高。注意，旧式疗法对应的块下方，表示死亡人数的矩形延伸到了新式疗法的块下面，而且不只延伸了一点点。这表明 death 和 anc 这两个变量是相关的（associated，或 correlated），也就是说旧式疗法组中的死亡比例大于新式疗法组中的死亡比例。然而正如我们之前指出的，相关性并不能证明因果关系。变量间也有可能存在更复杂的关系。要确定这一点，我们需要查看其他可获得的变量。

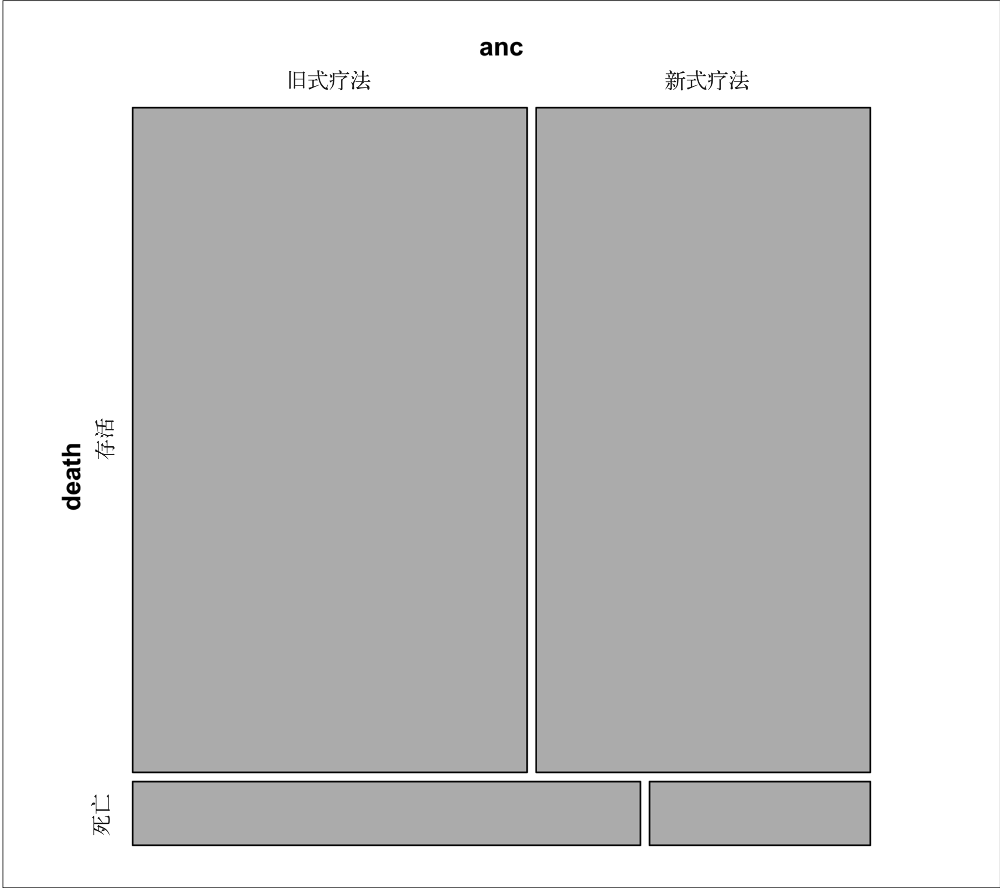


图 20-1：death（死亡率）和 anc（治疗频率）的马赛克图

马赛克图有助于研究两个分类变量，在研究三个变量时也会说明更多信息。接下来，我们将创建三向表，以便研究频率，然后把频率绘制成马赛克图。生成表的代码如下：

```
# 表20-2
xtab2 = table(clinic,death,anc)
xtab2 #生成下边的表
```

表 20-2：患者的治疗频率、死亡率和所在诊所

```
, , anc = old

      death
clinic  no yes
     A 176  12
     B 197  34

, , anc = new
```

```
       death
clinic  no yes
     A 293  16
     B  23   4
```

运行下面的代码，可以生成图 20-2：

```
# 图20-2
xtab2 = table(clinic,death,anc)
mosaic(xtab2)
```

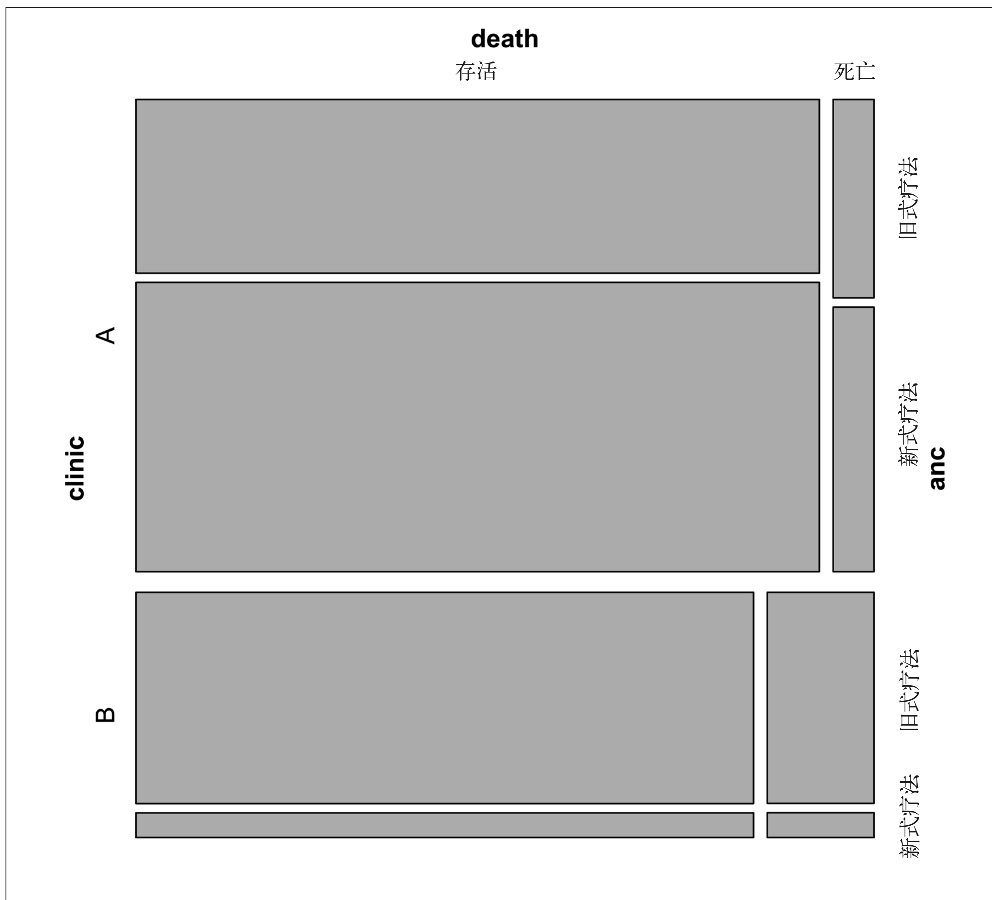


图 20-2：clinic（所在诊所）、death 和 anc 的马赛克图

我们从图 20-2 中可以找出一些十分有趣的关系。首先，图被分成了两部分，包括代表 A 诊所的上部和代表 B 诊所的下部。请单独查看每个部分。我们在双向表（见图 20-1）中看到的疗法和死亡之间的关系仍然存在于 A 诊所，而不存在于 B 诊所。总体考查整张表，B 诊所的死亡率比 A 诊所高很多，这是相当惊人的。值得注意的是，在 A 诊所有更多的女

性接受新式疗法，而在 B 诊所患者接受的都是旧式疗法。

这张图有几种修改方法，以突出某些关系。例如，假设我们想强调两家诊所疗法的分布有很大差异。我们可以给 `mosaic()` 命令再增加一个参数。代码如下：

```
# 图20-3
# 强调两家诊所不同疗法频率(anc)的分布差异
mosaic(xtab2, highlighting = "anc")
```

结果如图 20-3 所示。

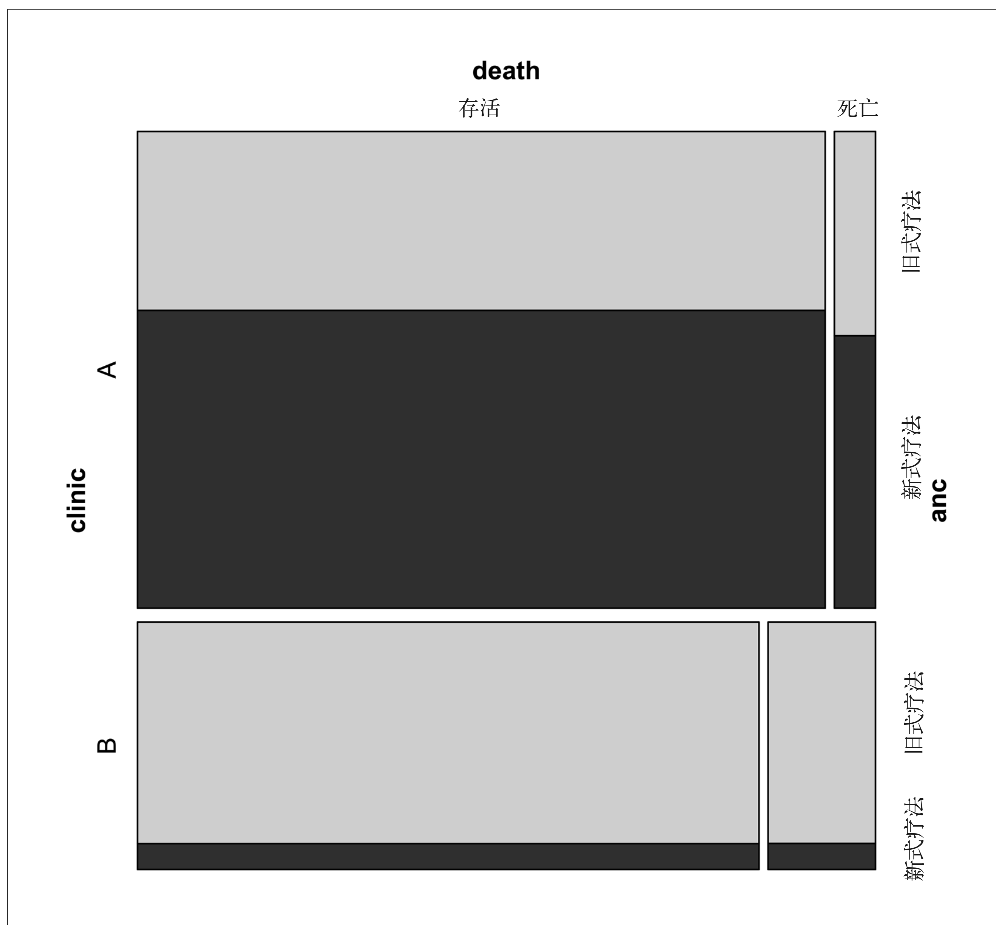


图 20-3：clinic、death、anc 的马赛克图，突出了 anc

我们还可以突出响应变量（response variable）。换句话说，这个变量是其他变量条件下的结果。在这里，响应变量是 `death`。我们也可以添加颜色向量，使图更有趣（你可能会发现，当块的颜色能形成鲜明对比时，这些关系更突出），代码如下：

```
# 图20-4
mosaic(table(clinic,anc,death),
  highlighting = "death",
  highlighting_fill = c("royal blue", "gold"))
```

结果如图 20-4 所示。

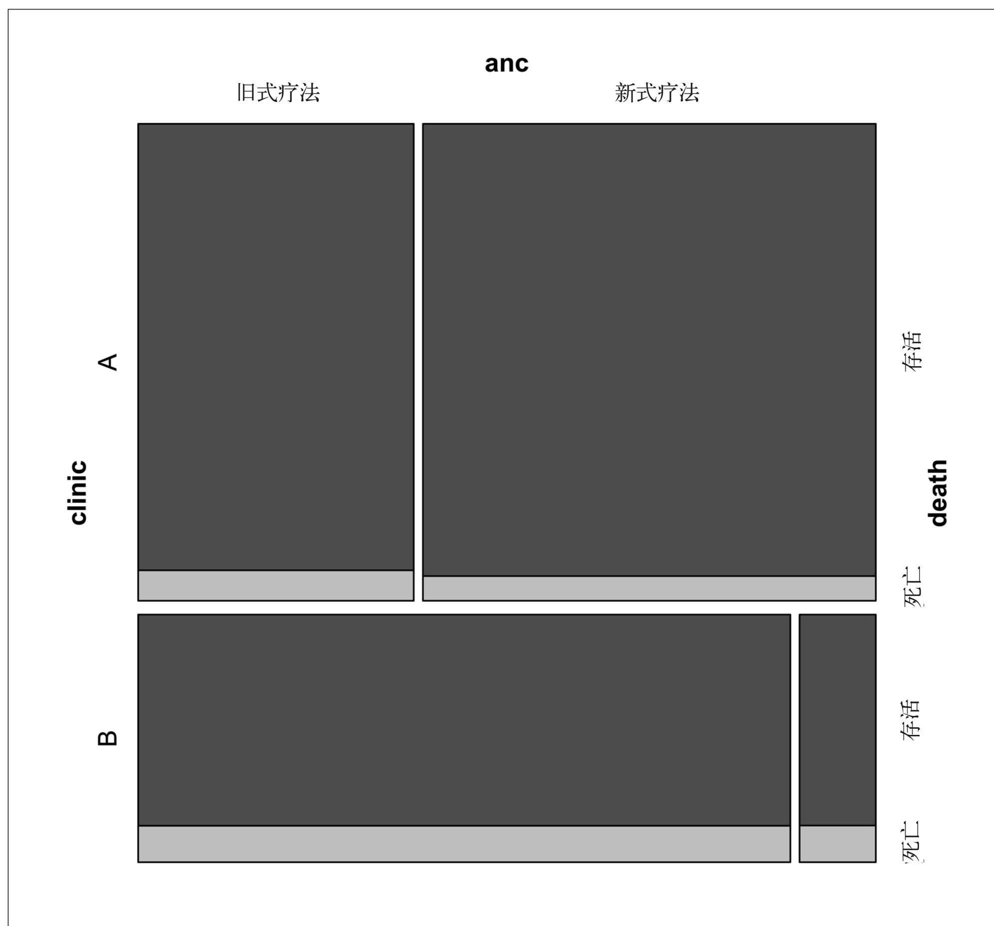


图 20-4：clinic、death、anc 的马赛克图，此处突出 death 并添加了颜色

有时改变表的方向可以突出或隐藏某些关联。你可以尝试突出不同变量，看看对结果的解释难度有何影响：

```
# 图20-5
mosaic(table(death,anc,clinic),
  highlighting = "clinic",
  highlighting_fill = c("royal blue", "gold"))
```

结果如图 20-5 所示。

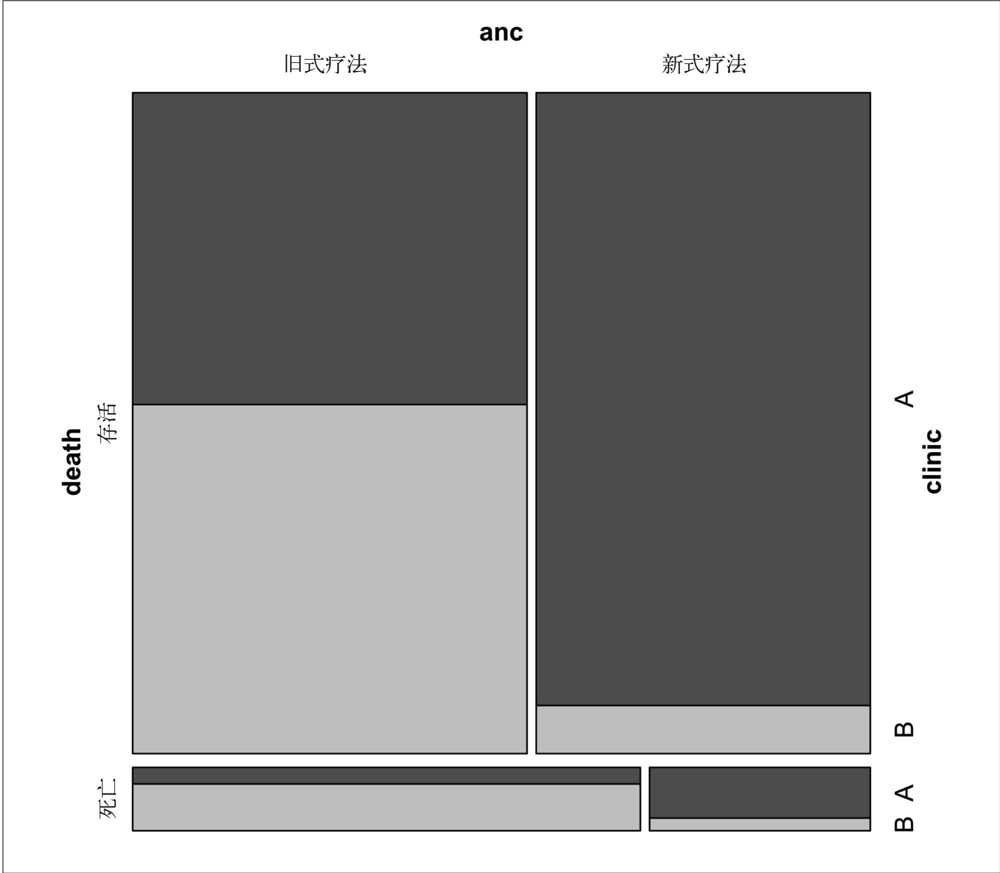


图 20-5：clinic、death、anc 的马赛克图，突出 clinic 并添加了颜色

解释马赛克图的另一种方法是*残差分析*（residual analysis）。如果表中的变量间没有关联，也就是说如果变量是*独立的*，我们可能会期望单元格是一定数量的。表中实际频率和预期值之间的差异被称为*残差*（residual）。下面显示了为双向表计算期望值的过程。

表 20-3：来自表 20-1 的实际值、计算总数、期望值和残差

```
      anc
death old new            Totals
  no  373 316          689 = 91.3% of 755
  yes  46  20           66 =  8.7% of 755

Expected values
     old                       new
no   382.5 (91.3% of 419)      306.6 (91.3% of 336)
yes  36.5 (8.7% of 419)        29.4  (8.7% of 336)

Residuals
```

```
       old                   new
no   373  -  382.5 = -9.5     316 - 306.6 = 9.4
yes   46  -  36.5  = 9.5       20 - 29.4  = -9.4
```

三向表也没有什么特殊的地方，但这里我们不做计算，因为计算有些棘手，而且就我们的目的而言，也没有计算的必要。马赛克图可以使用残差信息，展示实际频率和（在独立假设下的）预期频率之间的差异。要得到残差图，为 `mosaic()` 命令添加参数 `shade=TRUE` 即可。要解释图形上各种颜色和阴影的含义，则应当添加参数 `legend=TRUE`，代码如下：

```
# 图20-6
mosaic(xtab2, shade = TRUE, legend = TRUE)
```

结果如图 20-6 所示。蓝色块代表具有较大正残差的单元，而红色块代表具有较大负残差的单元。灰色块所代表的单元接近预期值。如果把 A 诊所中的四块和 B 诊所中相应的块比较一下，你会看到两家诊所之间的明显差异。

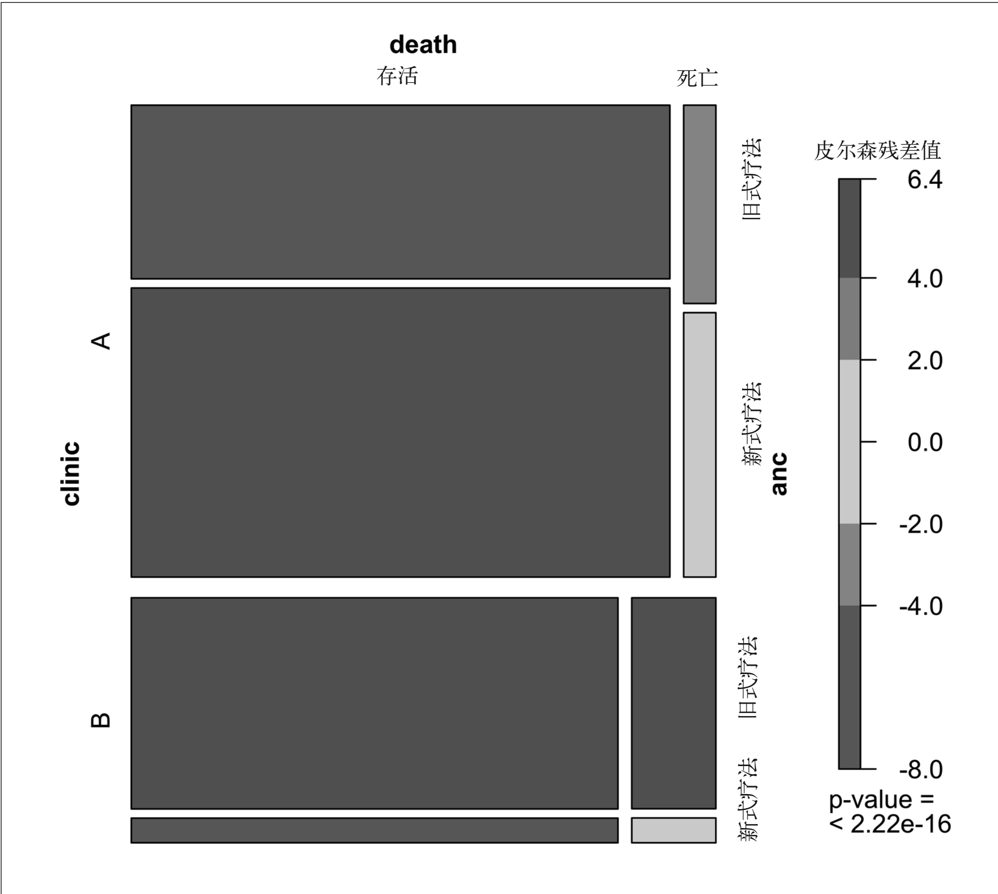


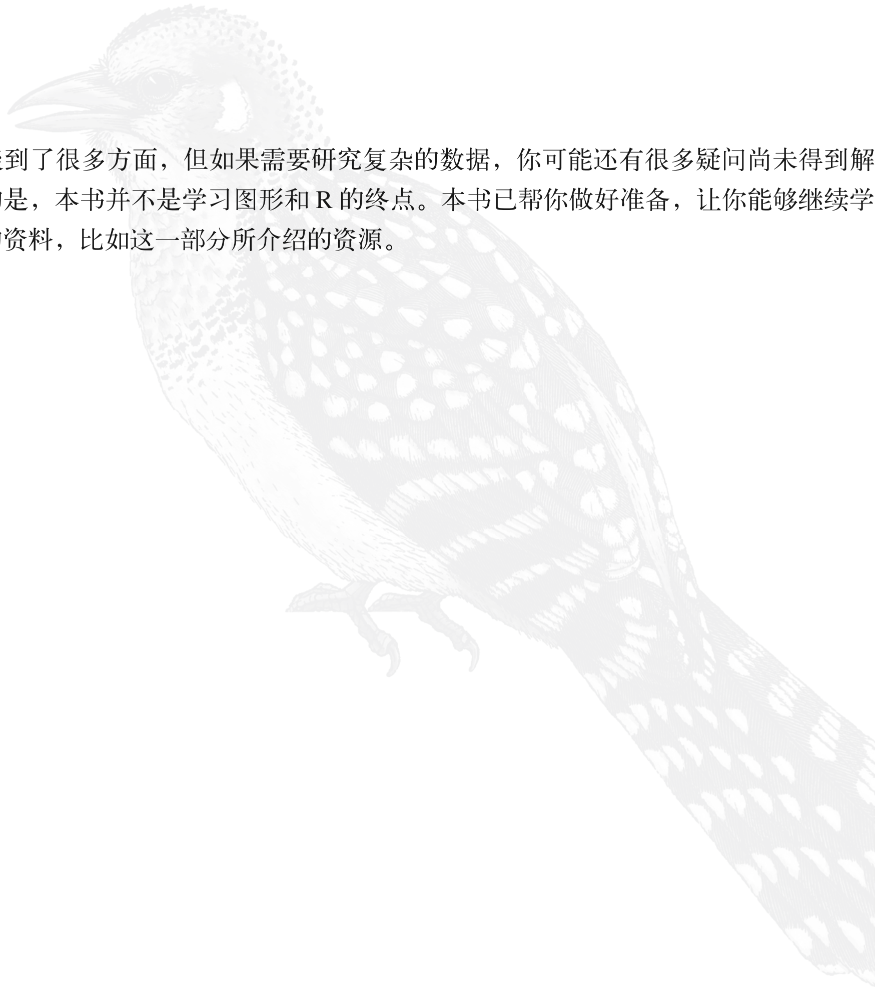
图 20-6：图 20-3 中结果的残差阴影（另见彩插）

练习20-1

众所周知的泰坦尼克号沉船事件一直是很多研究的主题。基础 R 安装自带一个关于生存者和死亡者的特征的数据集，就称为 `Titanic`。请使用马赛克图技术，确定哪些因素和生存相关。完成这项任务之后，请上网搜索，寻找至少两个使用同一个数据集的其他分析，将你的分析与之进行比较。这类研究应该不难找到。

第五部分

现在该做些什么



我们已经谈到了很多方面，但如果需要研究复杂的数据，你可能还有很多疑问尚未得到解答。幸运的是，本书并不是学习图形和 R 的终点。本书已帮你做好准备，让你能够继续学习更高级的资料，比如这一部分所介绍的资源。

第21章

拓展图形化知识和R技能的资源

艺术永无止境，除非放弃。

——列奥纳多·达·芬奇

关于R语言和图形的学习是永无止境的。本书谈不上面面俱到，但熟练掌握了本书内容的读者将来可以继续撰写这方面的著作。很多人设计出了实用的数据可视化方法，还有很多人为R贡献了自己的图形化方法实现，这一切令我十分惊叹。要想透彻介绍这些方法需要很多很多年，我的余生远远不够。当前可用的方法还没有介绍完，新的方法就又涌现了出来。这样的任务我心有余而力不足，因此，对图形化和R之真谛的进一步探索就交给你们了。

读了这本书后，你现在具备了使用R绘制图形并展示数据的基础知识。你学到的新技能有很多用途，并且毫无疑问，这些技能是你用得上的。当然，你也可能遇到以目前水平难以解决的问题。我希望本书能成为你的参考手册，帮助你解决许多问题，但如果想进一步学习，也有很多优秀的自学材料。下面列出的是我认为十分实用且便于理解的材料，在此推荐给你。完整的参考资料见附录A。

21.1 R图

CRAN（Comprehensive R Archive Network）是R项目（R Project）的一部分。CRAN任务视图（Task Views）概括了R包，并按照类别进行了整理，访问地址为http://cran.r-project.org/web/views/。单击“图”（Graphics），你会看到对图形包的一般性讨论，讨论提及了一些具体的包及其优势，还包括了许多图形包文档的链接。有些包可能有一些图形特

性，但本质上不算图形包，这里并没有提供这些包的信息。如果对这种包感兴趣，你可以看一看对应的类别。比如，如果你对存活曲线（survival curve）感兴趣，那就看看生存（Survival）类；如果你对绘制时间序列数据感兴趣，那就看看 TimeSeries 类；如果你对地理感兴趣，那就看看 SpacioTemporal 类。

如果你想钻研 `lattice` 包，此包的编写者 Sarkar（2008）的书非常值得一读。`ggplot2` 包的编写者 Wickham（2009）也写了一本易读的好书。Chang（2013）撰写了一本实用手册，介绍了很多在 R 中制图的方法，大多采用 `ggplot2` 包。如果你对 R 的基本知识和各种类型的图有所了解，那么这本书特别合适你，当然，你现在已经具备了这样的基础。

21.2　通用绘图原则

Tufte（1983）可能是在数据图形化方面被引用最多的书之一，其中收录了很多图形，归纳出了有效展示的诸多原则。书中还展示了许多正面的和反面的例子，你可以从中了解什么是好图形的特点。

Cleveland（1985，1993）的两本书逻辑严密，思维清晰，可以说是两本杰作，但要想完全理解，需要具备一点数学知识。大多数读者即使没有高等数学背景，也可以借助这本书更好地掌握绘图原则。相比现有的彩色展示，书中的图看起来有点平淡无奇，但 Cleveland 在图形设计方面超越了大多数人。许多 `lattice` 包都源自这两本书。

21.3　学习更多关于R的知识

现在，R 越来越受欢迎，市面上有大量关于 R 的图书。我说不出哪本最好，但我最喜欢的通用的 R 数据分析来自 Kabacoff（2011）。这本书刚刚出版经过扩展的第二版，我还没有拜读。要充分利用这本书，你应该了解基本的统计学知识。

如果想多了解 R 这门编程语言，请看看 Matloff（2011）。现在，你或许已经学习了大部分你想了解的关于 R 制图的内容，但是在数据处理、模拟、文本字符串方面仍有很多疑问，还有一些问题是你暂时没有想到的。对于这些，Matloff 都有所研究。

21.4　用R做统计

如果你在阅读本书之前没有任何统计学背景，你也许想了解一下这个领域。统计结合 R 的基础教材有几本，我推荐 Diez 等人（2012）的著作。为遵循 R 的开源理念，这本书是免费的，可以在 www.openintro.org 下载。纸质版在亚马逊网站上有售，价格低廉。书中使用的数据集在 `openintro` 包中。

我相信阅读本书是利用本章所讨论的大部分资源的前提，这也是我写作这本书的原因之

一。我希望这本书里的 R 作图知识能够对你有所帮助。

练习21-1

下面进行一项实战测试，看看你掌握了多少 R 技能。请重新生成图 1-3。

附录A

参考文献

Bland, J. M. and Altman, D. G. 1986. “Statistical methods for assessing agreement between two methods of clinical measurement.” *Lancet*, 327(8476) i: 307–10.

Boslaugh, Sarah. 2013. *Statistics in a Nutshell*, 2nd ed. Sebastopol, CA: O’Reilly.

Chang, Winston. 2013. *R Graphics Cookbook*. Sebastopol, CA: O’Reilly.

Cleveland, William S. 1985. *The Elements of Graphing Data*. Monteray, CA: Wadsworth.

———. 1993. *Visualizing Data*. Summit, NJ: Hobart Press.

de Vries, Andrie and Meys, Joris. 2012. *R for Dummies*. Chichester, England: John Wiley & Sons.

Deng, Henry and Wickham, Hadley. 2011. “Density estimation in R.” http://vita.had.co.nz/papers/density-estimation.pdf.

Diez, David M., Barr, Christopher D., and Çetinkaya-Rundel, Mine. 2012. *OpenIntro Statistics*, 2nd ed. www.openintro.org.

Few, Stephen. 2009. *Now You See It*. Oakland, CA: Analytics Press.

Fox, John. 2005. “The R Commander: A Basic-Statistics Graphical User Interface to R.” *Journal of Statistical Software*, 14(9): 1–42.

Gomez, M. and Hazen, K. 1970. “Evaluating sulfur and ash distribution in coal seams by statistical response surface regression analysis.” Report RI 7377, US Bureau of Mines, Washington, DC.

Hanneman, S. K. 2008. “Design, Analysis and Interpretation of Method-Comparison Studies.” *AACN Advanced Critical Care*, 19(2): 223–34.

Iannaccone, L. R. 1994. “Why Strict Churches Are Strong.” *American Journal of Sociology*, 99(5): 1180–211.

James, Gareth, Witten, Daniela, Hastie, Trevor, and Tibshirani, Robert. 2013. *An Introduction to Statistical Learning: with Applications in R*. New York: Springer.

Janert, Philipp K. 2011. *Data Analysis with Open Source Tools*. Sebastopol, CA: O'Reilly.

Kabacoff, Robert I. 2011. *R in Action*. Shelter Island, NY: Manning.

Kleinman, Ken and Horton, Nicholas J. 2014. *SAS and R: Data Management, Statistical Analysis and Graphics*, 2nd ed. Boca Raton, London, New York: CRC Press.

Ligges, U. and Maechler, M. 2003. “3D Scatter plots: an R Package for Visualizing Multivariate Data.” *Journal of Statistical Software* 8(11): 1-20.

Matloff, Norman. 2011. *The Art of R Programming*. San Francisco, CA: No Starch Press.

Murrell, Paul. 2011. *R Graphics*, 2nd ed. Boca Raton, FL: Chapman and Hall.

Ramsey, Fred and Schafer, Daniel. 2001. *Statistical Sleuth*, 2nd ed. Pacific Grove, CA: Brooks/Cole.

Sarkar, Deepayan. 2008. *Lattice: Multivariate Data Visualization with R*. New York, NY: Springer.

Tufte, Edward R. 1983. *The Visual Display of Quantitative Information*. Chesire, CT: Graphics Press.

Tukey, John W. 1977. *Exploratory Data Analysis*. Reading, MA: Addison-Wesley.

Wainer, Howard. 1984. “How to Display Data Badly.” *American Statistician*, 38(2): 137–47.

Wickham, Hadley. 2009. *ggplot2: Elegant Graphics for Data Analysis*. New York: Springer.

Wilk, M. B. and Gnanadesikan, R. 1968. “Probability plotting methods for the analysis of data,” *Biometrika*, 55(1): 1–17.

Wilkinson, Leland and Friendly, Michael. 2009. “The History of the Cluster Heat Map.” *American Statistician*, 63(2): 179–84.

Wong, Dona M. 2010. *The Wall Street Journal Guide to Information Graphics*. New York: W. W. Norton.

Yau, Nathan. 2011.*Visualize This*. Indianapolis, IN: John Wiley & Sons.

Yau, Nathan. 2013. *Data Points: Visualization That Means Something*. Indianapolis, IN: John Wiley & Sons.

附录B

R的颜色

使用如下命令，可以显示 657 种命名的 R 颜色：

```
> demo(colors)
```

使用如下命令，可以获取颜色名称列表：

```
> colors()
```

下面的脚本能够生成图 B-1 中的颜色表（如果你想打印自己的副本，可以直接复制代码）：

```
# 生成颜色表的脚本
par(col.axis="white",col.lab="white", mar=c(0.1,0.1,0.4,0.1),
  xaxt="n",yaxt="n")

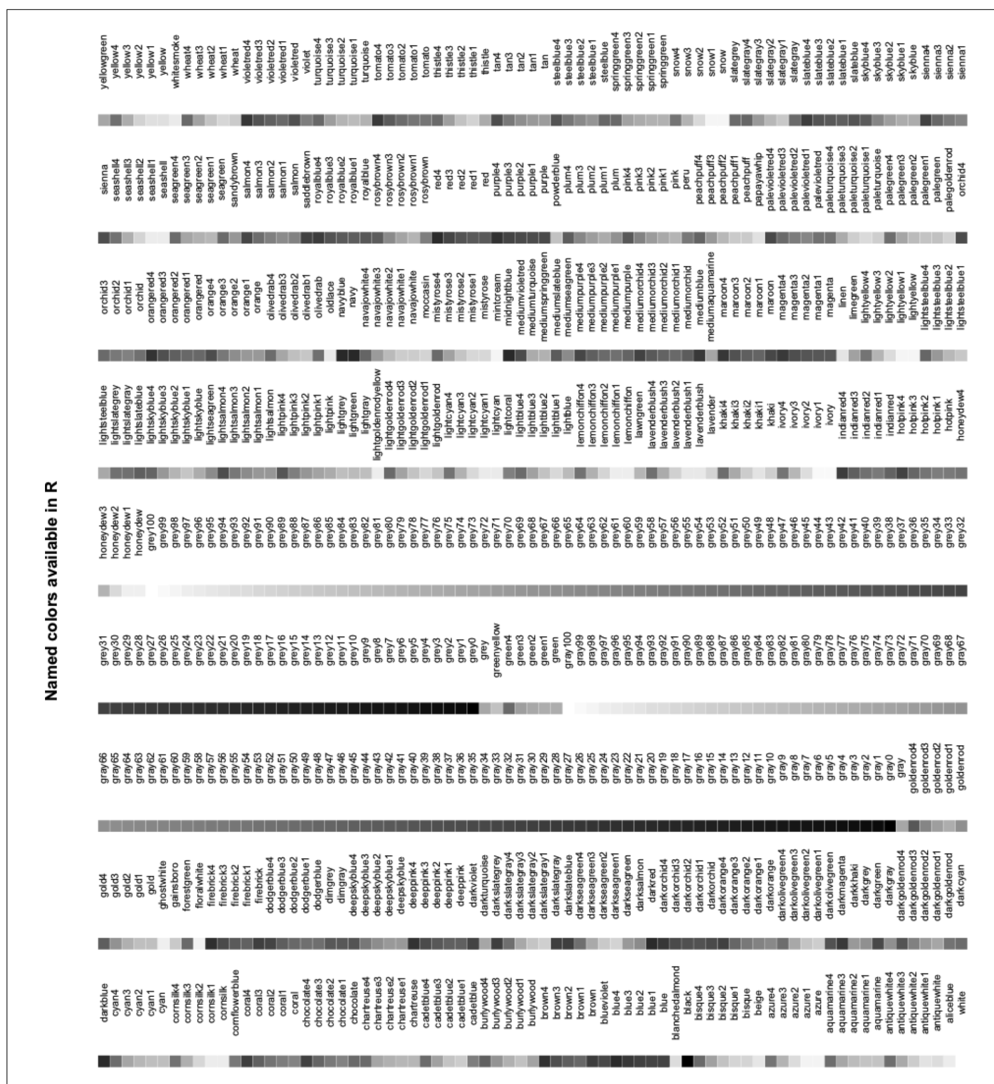
n = c(0:656) # 每种颜色的编号
n2 = (n %%73) # 每一列颜色有1~73个编号
cc = t(colors()) # 颜色名称
k = (2:9) # 每一列的编号
x=rep(c(1),times=73)

for(i in k) {
r = rep(c(i),times=73)
x = (c(x,r))
}

# 在(x,n2)处打印颜色矩形
plot(x,n2,col=cc,pch=15,
xlim=c(0,10),
ylim=c(0,73),
bty="n",
```

```
main="Named colors available in R",cex.main=.65)

x1 = x+ 0.5
text(x1,n2,cc,cex=.4)   # 在(x1,n2)处打印颜色名向量
```



B-1：657 种已命名的颜色（另见彩插）

你可以从网上找到哥伦比亚大学 Tian Zheng 教授制作的精美 R 颜色表，网址是 http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf。

附录C

R Commander图形用户界面

有些人就是不喜欢 R 的命令行界面，他们更愿意在图形用户界面（GUI，又名点击式界面）环境下工作。不常使用 R 的人很难熟记 R 的命令，要么总是输入错误，要么连绘制简单的图形也缓慢得令人痛苦。使用 R Commander 可以使你的生活更轻松，但需要说明的是，使用点击式界面时不能使用所有的 R 功能。

想尝试使用 R Commander，首先必须使用下面的命令安装它：

```
> install.packages("Rcmdr", dependencies=TRUE)
```

安装一次即可，但在使用它的每一个会话中，你都必须加载它。加载命令如下：

```
> library(Rcmdr)
```

图 C-1 所示为 R Commander 的窗口。你会发现，使用 R Commander 可以更快地生成常规图 / 表 / 分析，但无法生成一些高度定制的图形。如果想同时使用图形用户界面和命令行界面，你可以让控制台保持打开状态，然后在两个窗口之间来回切换。或者，你可以在 R Commander 的“R 脚本”（R Script）窗口输入一条命令并选中：单击命令所在行的开始，拖动到该行的末端，并释放鼠标按钮即可。这时选中的行会高亮显示。单击“提交”（Submit）按钮，R 将执行该命令。

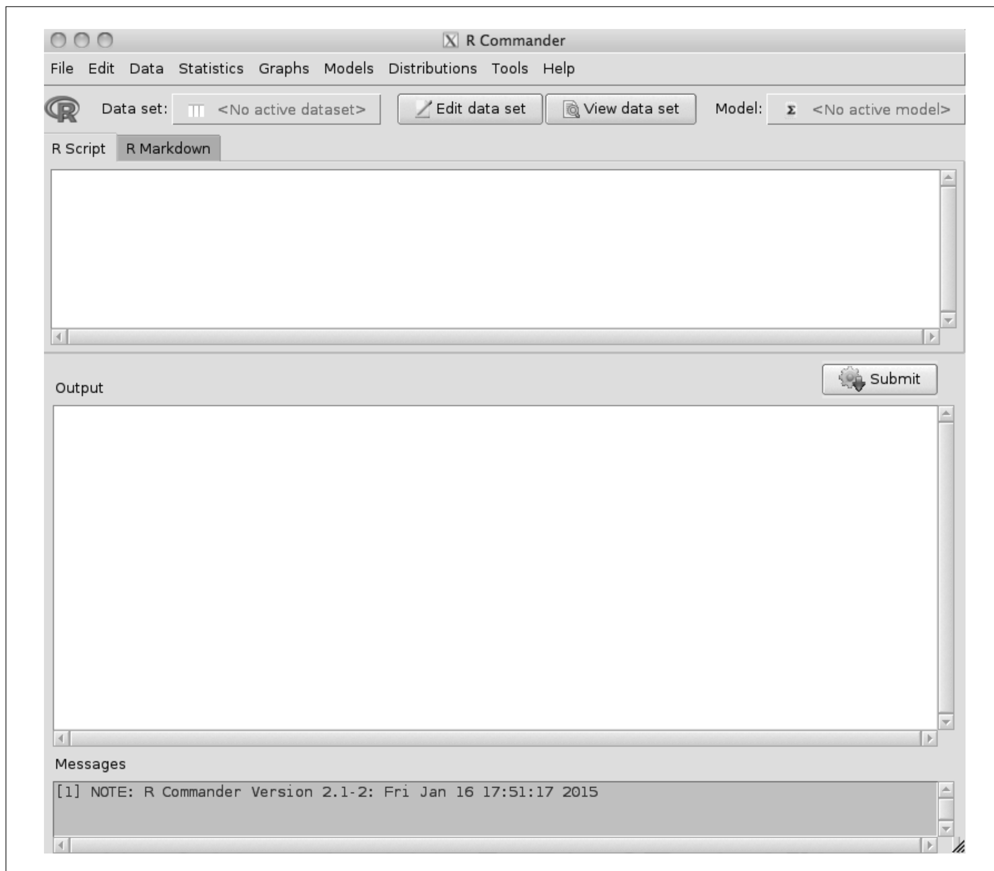


图 C-1：R Commander 的图形用户界面

请尝试使用 R Commander 解决第 3 章中的带状图问题。在屏幕顶部的菜单栏上，单击“数据”（Data）。在打开的菜单中，选择“包中的数据”（Data in packages）和“从附加包读取数据”（Read dataset from an attached package）。图 C-2 所示为打开的窗口，在其中可以选择 `trees` 数据集。然后点击“确定”（OK）。

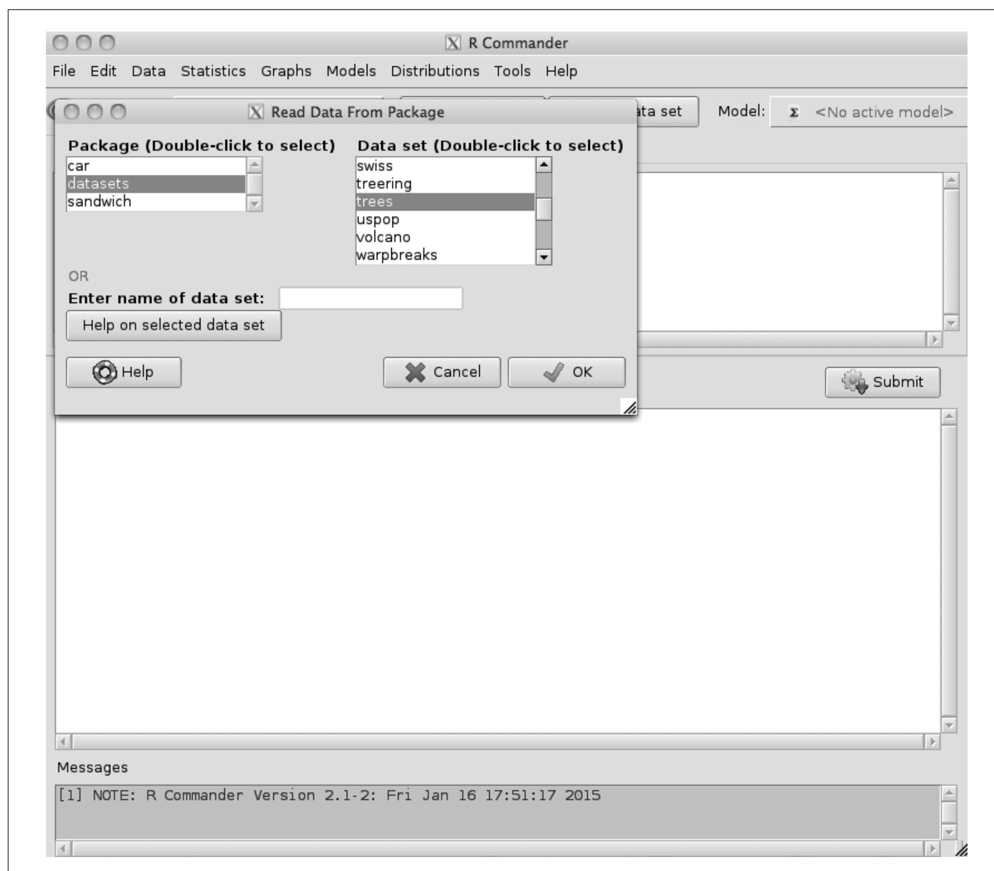


图 C-2：在 R Commander 中选择一个数据集以便用 R 分析

在菜单栏下面，你会看到 `trees` 是当前活跃的数据集。接下来，从图形（Graphs）窗口选择“带状图”（strip chart）。将你的结果与第 3 章中的第一张图进行比较。然后，尝试重现展示了跳动（jittering）的第二张图。[提示：在打开带状图窗口之后，请单击“选项”（options）。] 只使用图形用户界面，你将无法重现该章中其他的图。如果想生成第三张图，你可以在 R 脚本窗口中提交命令行。最简单的方法是编辑生成之前那张图的命令行，添加参数 `pch = 20` 并编辑参数 `xlab`（见图 C-3）即可。最后你只需要选中编辑完成的命令，点击“提交”。

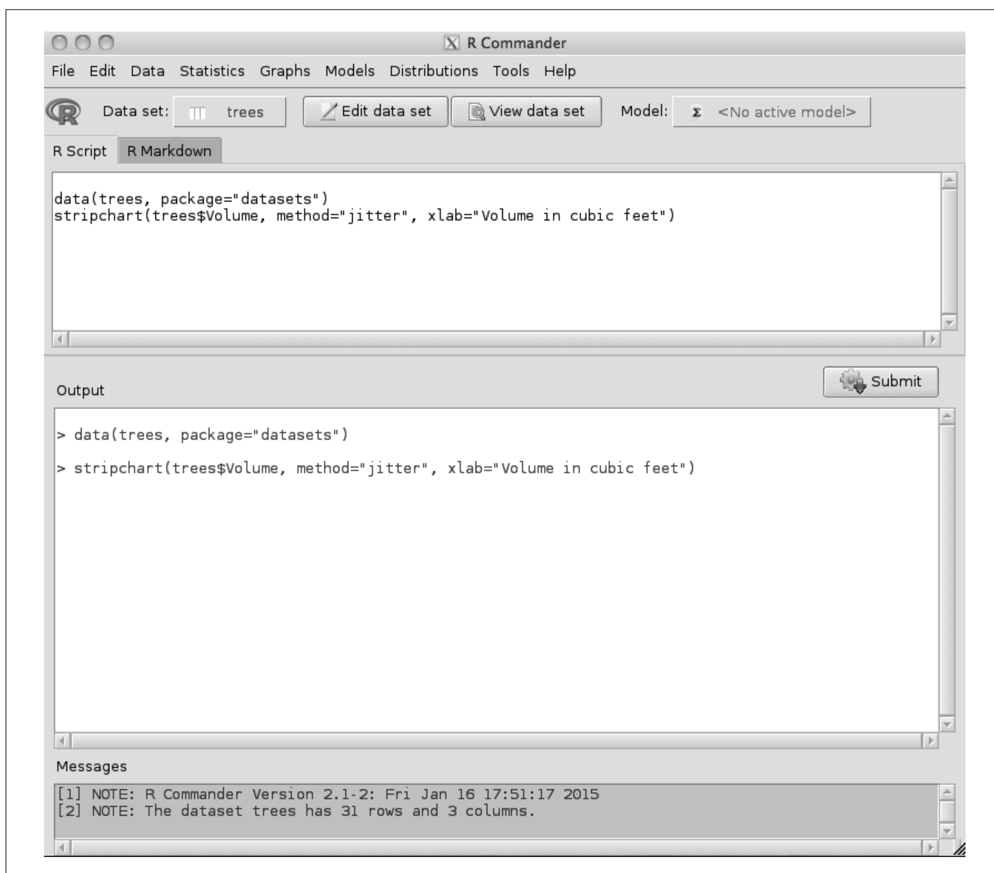


图 C-3：使用 R 脚本窗口把命令行提交给 R

如果你喜欢 R Commander，可以试着使用 GUI 重复实现本书中其他的例子。在很多情况下，如果你知道图应该是什么样子，就不难想出绘制办法。如果要绘制较复杂的图，那么你需要输入命令。

R Commander 可以扩展（即添加更多命令），你可以通过使用插件（plug-in）来实现。在我写作本书的时候，采用安装包的方式就可以安装 36 个可用的插件，它们大多数会添加很多新的命令。你也可以编写你自己的插件。想获得信息，请点击菜单栏上的"帮助"，或者查看 R Commander 网页（http://www.rcommander.com），或者访问作者的网页（http://socserv.mcmaster.ca/jfox/Misc/Rcmdr/）。你还可以访问 R 包的完整列表（http://cran.r-project.org/web/packages/available_packages_by_name.html），并向下滚动到以 RcmdrPlugin 开始的条目。

很多插件有一个或多个图形函数。你可以安装 RcmdrPlugin.HH，查看插件是如何工作的。HH 包含有许多实用的图形函数，R Commander 可以通过插件使用这些函数。首先，你需要安装包，代码如下：

```
> install.packages("RcmdrPlugin.HH", dependencies=TRUE)
```

接下来打开 R Commander，代码如下：

```
> library(Rcmdr)
```

打开 R Commander 的屏幕后，你可以在菜单栏点击“图形”（Graphs），看看没有插件时有多少选项。点击菜单栏的“工具”选项，并选择“加载 Rcmdr 插件”（Load Rcmdr plug-ins）即可加载插件。你需要在随后出现的菜单上选择插件的名称。现在，再看一下“图形”选项，你会注意到选项比之前更多了。增加的选项都在菜单的下半部分，其中一些确实很实用。例如，Scatterplot.HH 插件提供了更大的输出控制（你可以借此控制图符的类型和大小），提供了可以放置在图形上的几种线，你甚至可以通过点击识别一个点的功能。

其他几个插件也包括了良好的图形函数。与 RcmdrPlugin.HH 不同，一些插件会在菜单栏中添加新的菜单。添加有趣图形的插件，包括 RcmdrPlugin.KMggplot2、RcmdrPlugin.NMBU、RcmdrPlugin.EZR 等。

附录D

使用/引用的包

包	作者	描述
`Amelia`	James Honaker Gary King Matthew Blackwell	处理缺失数据的程序
`aplpack`	Hans Peter Wolf Uni Bielefeld	另一个绘图包（Another Plot PACKage）：添加了 `stem.eaf()`、`bagplot()`、`faces()`、`spin3R()`、`plotsummary()`、`plothulls()` 和一些滑块函数
`car`	John Fox Sanford Weisberg	应用回归伴侣（Companion to Applied Regression）
`corrplot`	Taiyun Wei	相关矩阵的可视化
`DescTools`	Andri Signorell 以及其他贡献者	描述性统计工具
`epade`	Andreas Schulz	简单图表（Easy Plots）
`epicalc`	Virasakdi Chongsuvivatwong	流行病学计算器
`foreign`	R Core Team	读取 Minitab、S、SAS、SPSS、Stata、Systat、Weka、dBase 等存储的数据
`GGally`	Barret Schloerke Jason Crowley Di Cook Heike Hofmann Hadley Wickham Francois Briatte Moritz Marbach Edwin Thoen	`ggplot2` 的扩展

（续）

包	作者	描述
ggplot2	Hadley Wickham Winston Chang	用于图表语法的实现
gmodels	Gregory R. Warnes 以及其他贡献者	用于模型拟合的各种 R 编程工具
gpairs	John W. Emerson Walton A. Green	用于生成广义的 pairs（gpairs）图
gplots	Gregory R. Warnes Ben Bolker Lodewijk Bonebakker Robert Gentleman Wolfgang Huber Andy Liaw Thomas Lumley Martin Maechler Arni Magnusson Steffen Moeller Marc Schwartz Bill Venables	用于数据绘图的各种 R 编程工具
grid	Paul Murrell	网格图形包
hexbin	Dan Carr 以及其他贡献者	六边形分组程序
HistData	Michael Friendly Stephane Dray Hadley Wickham James Hanley Dennis Murphy	来自统计和数据可视化历史的数据集
Hmisc	Frank E. Harrell Jr. 以及其他贡献者	Harrell 杂项（Harrell Miscellaneous）
lattice	Deepayan Sarkar	点阵图
latticeExtra	Deepayan Sarkar Felix Andrews	基于点阵图的附加的图形化实用工具
multcomp	Torston Hothorn Frank Bretz Peter Westfall 以及其他贡献者	广义参数模型中的联立推断
ncdf	David Pierce	Unidata netCDF 数据文件的界面
nlme	Jose Pinheiro Douglas Bates 以及其他贡献者	线性和非线性混合效果模型

（续）

包	作者	描述
plotrix	Jim Lemon、Ben Bolker, Sander Oom, Eduardo Klein、Barry Rowlingson、Hadley Wickham、Anupam Tyagi、Olivier Eterradossi、Gabor Grothendieck、Michael Toews、John Kane、Rolf Turner、Carl Witthoft、Julian Stander、Thomas Petzoldt、Remko Duursma、Elisa Biancotto、Ofir Levy、Christophe Dutang、Peter Solymos、Robby Engelmann、Michael Hecker、Felix Steinbeck、Hans Borchers、Henrik Singmann、Ted Toal、Derek Ogle	各种绘图函数
psych	William Revelle	心理学、心理测量学和性格研究的过程
Quandl	Raymond McTaggart Gergely Daroczi	Quandl 数据连接
Rcmdr	John Fox Milan Bouchet-Valat 以及其他贡献者	基于 tcltk 包且独立于平台的 R 基础统计 GUI
RcmdrMisc	John Fox 以及其他贡献者	R Commander 杂项函数
RcmdrPlu gin.EZR	Yoshinobu Kanda	EZR（Easy R）包的 R Commander 插件
RcmdrPlugin.HH	Richard M. Heiberger Contributions from Burt Holland	支持 HH 包的 Rcmdr
RcmdrPlu gin.KMggplot2[1]	Triad sou Kengo Nagashima	Kaplan-Meier 图的 Rcmdr 插件和使用 ggplot2 包绘制的其他图
RcmdrPlu gin.NMBU[2]	Kristian Hovde Liland Solve Sæbø	NMBU 统计的 R Commander 插件
ResearchMe-thods	Mohamed Abdolell Sam Stewart Hadley Wickham	使用 GUI 给非统计专业的学生讲授统计学 灵活改变数据形状：reshape 包的 reboot
rgl	Daniel Adler Duncan Murdoch 以及其他贡献者	使用 OpenGL 的三维可视化
scatterplot3d	Uwe Ligges Martin Maechler Sarah Schnackenberg	三维散点图
RCurl	Duncan Temple Lang	R 的通用网络（HTTP/FTP/……）客户端界面
Sleuth2	F. L. Ramsey D. W. Schafer 以及其他贡献者	来自 Ramsey 和 Schafer 的数据集（2001）

注 1：参见 https://cran.r-project.org/web/packages/RcmdrPlugin.KMggplot2/index.html。
注 2：参见 https://cran.r-project.org/web/packages/RcmdrPlugin.NMBU/index.html。

（续）

包	作者	描述
`sm`	Adrian Bowman Adelchi Azzalini	非参数回归和密度估计的平滑方法
`vcd`	David Meyer Achim Zeileis Kurt Hornik 以及其他贡献者	分类数据可视化
`XLConnect`	Mirai Solutions GmbH Martin Studer 以及其他贡献者	R 的 Excel 连接器
`XML`	Duncan Temple Lang	读取和创建 XML（和 HTML）文件

附录E

从R的外部导入数据

E.1 一些有帮助的网络数据仓库

你可以从很多网站中下载数据集，并用 R 进行数据分析。下文以列表的形式呈现了一些资源，其中列出的只是诸多共享数据的一小部分。在很多情况下，要使用数据集，你需要注册并同意使用条款。在获取数据之前，你应当仔细阅读供应商的要求。下面这些资源的数据集经常以 Excel 或 CSV 格式提供，这一点已经在 1.8.3 节中讨论过。其他格式的例子如下。

开放访问目录（Open Access Directory，http://oad.simmons.edu/oadwiki/Main_Page）

该网站提供可下载的数据的链接，数据关乎各种主题，科研数据居多。许多数据集是免费的，有些必须购买。向下滚动内容目录到“数据仓库”，可以查看所涵盖的主题。

FedStats（http://fedstats.sites.usa.gov/）

这是美国联邦政府的数据库，包含各种免费开放的数据。此网页上有各政府机构共享数据的链接。

DATA.GOV（http://catalog.data.gov/dataset）

这是另一个联邦数据仓库。

Quandl（http://www.quandl.com）

这是一个拥有 1000 万余个数据集的仓库，可以提供多种格式的免费下载，包括 R 数据框。相比许多其他来源，Quandl 更加易于操作。安装并加载 Quandl 包的代码如下：

```
> install.packages("Quandl")
> library(Quandl)
```

你可以浏览 Quandl 网页，找到想要的文件。例如，假设你在 http://www.quandl.com/FBI_UCR/ USCRIME_STATE_PENNSYLVANIA 这个网页上选中了美国联邦调查局宾夕法尼亚州的“州犯罪”（Crimes by State）文件，你可以用一条命令把它加载到 R 数据框 penn.crime，代码如下：

```
> penn.crime = Quandl("FBI_UCR/USCRIME_STATE_PENNSYLVANIA")
```

E.2 把各种类型的数据导入R

R 可以读取很多不同格式的数据。本节将讨论如何导入一些最重要格式的数据。

E.2.1 CSV

我们的第一个例子是一个来自美国国家科学基金会的简单的 CSV 文件。注意，它似乎和 1.8.3 节中的例子非常像。但该文件不在你计算机的工作目录中，所以你必须在引号内包括整个 URL，表明它来自哪个网页，代码如下所示：

```
> nsf2011 = read.csv(
  "http://www.nsf.gov/statistics/ffrdc/data/exp2011.csv",
    header=TRUE)
```

E.2.2 统计软件包（SPSS、SAS等）

在宗教协会数据档案（Association of Religion Data Archives，ARDA，http://www.thearda.com）中，我发现了一个有趣的数据集。读完 ARDA 之后，在页面顶部的菜单栏上点击“归档数据”（Data Archive）即可查看可获得哪些数据集。这里的数据集有很多种不同的格式。例如，从 http://www.thearda.com/Archive/Files/Downloads/CEMFILE_DL.asp 可以下载由 Wilbur Zelinsky 搜集的“墓碑索引”（The Gravestone Index），有三种版本可选。其中，SPSS 和 STATA 两种格式分别为两种有竞争关系的统计软件包设计。一个名为 foreign 的 R 包可以将这些格式转换为 R 的数据框。安装和加载包的代码如下：

```
> install.packages("foreign", dependencies=TRUE)
> library(foreign)
```

下载 SPSS 文件到工作目录以后，通过使用如下命令，可以将其读取到以 stone 命名的 R 数据框：

```
> stone=read.spss("The Gravestone Index.SAV",to.data.frame=TRUE)
> fix(stone)  # 看编辑器中的数据
```

`foreign` 包也可读写其他数据格式，比如 Minitab、SAS、Octave 和 Systat。想进一步了解 `foreign` 包，你可以采用如下命令：

```
> library(help=foreign)
```

E.2.3 ASCII

墓碑文件（Gravestone file）也有固定宽度格式（fixed-width format）的 ASCII 文件。其数据落在每一行的固定位置上，在数据点之间没有空格或其他分隔符。前几行和最后几行的内容如下：

```
11862  8 11820000000000000000000000000000000000
11868  8 11820000000000010000000000000000000000
11875  8 11820000000000000000000000000000000000
11910  8 11820000000000000000000000000000000000
11885  8 11820000000000000000000000000000000000
11861  8 11820000000000010000000000000000000000
11864  8 11820000000000010000000000000000000000

52003 18 64120000 0 110 0 00000001 0 0000000000 0
52003 18 64120000 0 1 0 0 00000000 0 0000000000 0
52007 18 64120000 0 0 0 0 0000000010 0010000000 0
52003 18 64120000 0 0 0 0 00000000 0 0000000000 0
51990 18 64120000 0 0 0 0 00000000 0 0000000000 0
```

码本的第一部分如下：

```
1) BOOKNUM: 1
2) YBIRTH: 2-5
3) CEMNAME: 6-8
4) YEAREST: 9-10
5) CITYCEM: 11-12
6) COUNTRY: 13
7) COLLYEAR: 14
8) GOTHICW: 15
9) MARRIAG: 16
10) HEART: 17
11) HEARTSS: 18
12) MILITAR: 19
13) SECMESS: 20
14) OCCUPAT: 21
```

可以使用 `read.fwf()`（读取固定宽度格式）命令读取数据。注意，这里没有头标信息。若包含参数 `header=TRUE`，则 R 将受到误导，试图根据第一行的数字分配变量名。这可能会导致错误信息。参数 `widths` 是必须包含的，后面需要跟着一个向量，给出每一个变量的列宽，如码本中所示。第一个变量 `BOOKNUM` 有 1 列；第二个变量 `YBIRTH` 有 4 列（第 2~5 列）；第三个变量 `CEMNAME` 有 3 列（第 6~8 列）；以此类推。下列命令读取已经复制到工作目录的 ASCII 文件：

```
gs = read.fwf("The Gravestone Index.DAT",widths = c(1,4,3,2,2,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1))
```

或者，不必输入 42 个 1，你可以使用 `rep()` 函数完成相同的功能，代码如下：

```
> gs = read.fwf("The Gravestone Index.DAT",
    widths = c(1,4,3,2,2,rep(1,42)))
```

相同数据的 SPSS 数据文件有变量名，但 ASCII 文件没有名字，并且分配给变量的名字为 `V1`、`V2`、`V3` 等。你可以创建一个新向量，为变量赋予来自码本的真实名字，代码如下：

```
vars = c("BOOKNUM", "YBIRTH", "CEMNAME", "YEAREST", "CITYCEM",
"COUNTRY", "COLLYEAR", "GOTHICW", "MARRIAG", "HEART", "HEARTSS",
"MILITAR", "SECMESS", "OCCUPAT", "PICTORIA", "DECEAEL",
"PHOTODEC", "RELMES", "SYMBOL", "ANGEL", "SYMBOOK", "SYMDEATH",
"DOVE", "FISH", "FINGERS", "SYMDIVIN", "GATES", "HANDSIP",
"IHSE", "HANDS", "LAMB", "SYMCROSS", "STATUE", "STAANGEL",
"STABOOK", "STADIVIN", "STALAMB", "STACROSS", "EFFIGY",
"WEEPWIL", "SECULAR", "SYMCHURC", "HANDSCIP", "CROWN",
"STADOVE", "PICCHURC", "STADEATH")
```

然后可以在向量 `vars` 中给出 `gs` 变量的名字：

```
> names(gs) = vars
```

E.2.4 XML

XML（Extensible Markup Language，可扩展标记语言）是一种文本格式，用于数据交换。数据有多种不同格式，其中一些是专有的，甚至是保密的，因此把每一种格式都转换成其他所有格式几乎是不可能的。XML 透明而开放，是在不同计算机系统和应用间共享数据的常用方式。R 有一个 `XML` 包，R 用户可以借此读取和创建 XML 文件。在网页 http://www.omegahat.net/RSXML/上可以找到该包的文档。XML 文件可能比目前为止我们看到的平面文件复杂得多。在将 XML 转换为 R 数据框之前，通常需要对其进行一些探索，以了解它的结构。下面是将一个相对简单的 XML 文件转换为数据框的例子。这是联邦选举委员会 2009—2010 候选人概要文件，详见 http://catalog.data.gov/dataset/2009-2010-candidate-summary-file。

安装并加载 `XML` 包之后，就可以执行转换了，代码如下：

```
> install.packages("XML",dependencies=TRUE)
> library(XML)
> cand = xmlToDataFrame("CandidateSummaryAction.xml")
```

E.2.5 netCDF

在国家冰雪数据中心（National Snow and Ice Data Center，http://nsidc.org）的数据仓库列表中，你可以找到下面的数据集。我选择用它来展示另一种数据类型，即 netCDF（network Common Data Form，网络公共数据格式）文件。该格式因存储大量的科学数据（特别是地球物理数据）阵列而广为使用。和 XML 一样，此格式的数据集比较复杂。你可以使用 FTP 从 https://gigclerk.com/seoclerks/122001/get-201-high-pr-awesome-backlinks-seo-links-to-any-website-blog-twitter-facebook-page-wikis-pinterest-youtube-videos-ins 下载数据集并将其保存到工作目录中。以下代码可以安装并加载 `ncdf` 包，在 R 中处理这些数据：

```
> install.packages("ncdf")
> library(ncdf)
```

这个数据集是一个相当复杂的对象列表，每一个对象本身都是一个对象列表。按 netCDF 的说法，每一个主列表是一个“变量”。为了在 R 中使用数据，你必须定义数据的一个子集，该数据将包括与变量相关的项的列表。你可以使用如下方式实现上述操作：

```
> ice = open.ncdf("seaice_conc_monthly_nh_f08_198707_v02r00.nc")
> # 创建一个命名为"ice"的R对象
> str(ice) # 表明ice是由其他列表组成的列表
> icedata = get.var.ncdf(ice,"seaice_conc_monthly_cdr")
> close.ncdf(ice)
```

`str(ice)` 命令的结果中可以找到变量的名字，`seaice_conc_monthly_cdr` 是为本例所选的名字。在大多数情况下，为选择一个变量名，你需要对数据有更多的了解。

E.2.6 网页抓取

包含在网页中的数据也可以复制。这通常被称为*网页抓取*（Web scraping）。本书不会详细讨论这个话题，如果你有提取网页数据的需求，可以从 `download.file()` 和 `readLines()` 的帮助文件入手。`RCurl` 和 `XML` 等包也许能派上用场。

附录F

章节练习解答

本书为每个练习提供了答案。请先认真解答习题，然后再来看答案。在 R 中，很多问题会有多种解决方案。如果你想出的方案和答案不同，试着看看这两个解决方案是否等同——两个方案的结果是否相同？为什么相同，为什么不同？

F.1　练习1-1到练习1-4

答案见第 1 章。

F.2　练习3-1

```
attach(mtcars)
stripchart(mpg ~ cyl, method = "jitter")
```

这有助于将汽车分开一点。现在，我们可以看出每组有多少汽车了。

不出所料，气缸较少的汽车油耗也较少。

F.3　练习3-2

```
install.packages("plotrix", dependencies=TRUE)
library(plotrix)
attach(trees)
dotplot.mtb(Volume)
```

有一种抖动是自动的。即便如此，一些非常近的值仍会跑到一块儿。解决此问题的方法之一是缩小图符，代码如下：

```
dotplot.mtb(Volume, pch = 20) # 或者
dotplot.mtb(Volume, pch = ".") # 太小!
dotplot.mtb(Volume, pch = "/") # 嗯……
detach(trees)
```

F.4 练习4-1

```
dotchart(USArrests$Murder, labels = row.names(USArrests))
```

州的名字太大，产生了重叠，变得模糊不清。

F.5 练习4-2

```
load("Nimrod.rda") # .rda说明被保存为R数据框
dotchart(Nimrod$time)
```

好！

```
dotchart(Nimrod$time, labels = Nimrod$performer, cex = .5)
```

更好！

```
Nimrod2 = Nimrod[order(Nimrod$time),]
dotchart(Nimrod2$time, labels = Nimrod2$performer, cex = .5)
```

好极了！

F.6 练习5-1

```
# 在屏幕打印结果
library(nlme)
attach(MathAchieve)
boxplot(SES ~ Minority * Sex)

# 图片到文件
pdf("SES.pdf") # 打开一个装置
library(nlme)
attach(MathAchieve)
boxplot(SES ~ Minority * Sex)
dev.off() # 关闭并保存文件
```

把 SES.pdf 文件插入文字处理程序文档。

SES 与 Minority 和 Sex 的关系，似乎同 MathAch 与 Minority 和 Sex 的关系一样。

F.7 练习5-2

```
par(mfrow = c(1,2)) # 并列展示2张图
attach(mtcars)
boxplot(mpg ~ cyl)
library(plotrix)
ehplot(mpg, cyl)
detach(mtcars)
par(mfrow=c(1,1)) # 重置为1张图片/页面
```

箱线图展示了参考点（即四分位）；EH 图展示了实际点，包括抖动（jittering）。

你可以为 EH 图添加盒子和胡须：`ehplot(mpg, cyl, box = T)`。

F.8 练习6-1

```
library(multcomp)
stem(sbp$sbp, scale = .5)
stem(sbp$sbp, scale = 3)
stem(sbp$sbp, scale = 4)
```

选择大于 3 的 `scale` 得到的图似乎毫无价值。不过，其他数据集未必如此。

F.9 练习6-2

```
install.packages("aplpack", dependencies = T)
library(aplpack)
attach(trees)
stem.leaf.backback(Height, Volume)
detach(trees)
```

测量单位不同。如果将单位标准化，会发生什么？

```
library(car)
attach(Baumann)
stem.leaf.backback(pretest.1, post.test.1)
```

似乎期末测试的分数更高。

F.10 练习7-1

```
library(Sleuth2)
attach(case0302)
par(mfrow = c(2,2)) # 一个页面展示4张图片
hist(Dioxin)
hist(Dioxin, breaks = 20)
hist(Dioxin, breaks = 30)
```

```
hist(Dioxin, breaks = 40)
par(mfrow = c(1,1)) # 重置为1张图片/页面
```

分布形状变化很大。没有指定中断时，大多数点看来聚集在零附近。有很多断点时，添加了几个极值，分布看起来几乎是对称的。带状图强化了这个观点。看来仅有两点是不符合规则的。它们为什么如此不同？这两个点是不是错的？应该将它们包括在内吗？

F.11　练习7-2

```
library(Hmisc)
library(car)
attach(Burt)
histbackback(IQbio, IQfoster)
detach(Burt)
```

两个直方图看起来非常相似，只是 `IQbio` 组有几个更高的智商值。

`Salary` 数据集中男性和女性的工资比以前的问题更难研究，因为没有单独的“男性工资”和“女性工资”向量。因此，为了使用 `histbackback()`，你必须创建这样的向量。做到这一点有几种方法，如下代码是其中之一：

```
attach(Salaries)
m = subset(Salaries, sex == "Male") # 只包含男性数据
f = subset(Salaries, sex == "Female") # 只包含女性数据
histbackback(m$salary, f$salary)
detach(Salaries)
```

F.12　练习8-1

```
library(multcomp)
eq2 = density(sbp$sbp, bw = 4) # 图8-1c
hist (sbp$sbp,
  main="c. Histogram + Kernel Density, bw = 4",
  col = "maroon", las = 1, cex.axis = .8,
  freq = F) # freq=F: 概率密度
lines(eq2,lwd = 2) # 在已有的直方图上绘制密度曲线
```

这里比 `bw = 5` 时弯曲得更剧烈。

```
eq2 = density(sbp$sbp, bw = 2) # 图8-1c
hist (sbp$sbp,
  main="c. Histogram + Kernel Density, bw = 2",
  col = "maroon", las = 1, cex.axis = .8,
  freq = F) # freq=F: 概率密度
lines(eq2,lwd = 2) # 在已有的直方图上绘制密度曲线
```

以上生成了十几处弯曲，细节并不真正符合直方图。不过，直方图本身就是非常粗略的近似。

```
eq2 = density(sbp$sbp, bw = 20) #图8-1c
hist (sbp$sbp,
  main="c. Histogram + Kernel Density, bw = 20",
  col = "maroon", las = 1, cex.axis = .8,
  freq = F) # freq=F: 概率密度
lines(eq2,lwd = 2) # 在已有的直方图上绘制密度曲线
```

这条线比 bw = 10 时平坦，但变化并不十分明显。

F.13 练习8-2

sbp 为 125 时，*y* 坐标大约在 0~0.25 的中间位置，即 0.125 上下。也就是说，选择一个收缩压为 125 或更小值的患者的概率约为 12.5%。

通过类似的处理，我们可以确定小于等于 175 的收缩压的概率约为 90%。选择收缩压大于等于 175 的人的概率是 1 − prob[175 或更小] = 1 − 0.9 = 0.1，即 10%。

y 坐标落到 125~175 的概率是 prob[175 或更小] − prob[125 或更小] = 90% − 12.5% = 77.5%。

F.14 练习9-1

Male 条在每一组的上方，图例的顺序是相反的，看起来令人迷惑。把 legend() 中的 sexlab 改为 c("Male", "Female")。

F.15 练习9-2

```
library(car)
attach(Salaries)
library(epicalc)
salK=salary/10000
pyramid(salK,sex, binwidth = 1,
  col = "seagreen",
main = "Salaries in 10,000s of Dollars",
cex.bar.value =.4, cex.axis = .8)
```

请将此例和练习 7-2 的结果进行对比。

F.16 练习10-1

```
install.packages("HistData")
library(HistData)
attach(Nightingale)
deaths = c(sum(Disease), sum(Wounds), sum(Other))
pie(deaths, labels=c("Disease","Wounds","Other"))
```

deaths 命令在每一列中找到总值，这些列包括 Disease、Wounds 和 Other。找到的总计用来绘制饼图。Disease 和其他原因之间的区别非常明显，但其他变量之间的差异不易看到。

F.17　练习10-2

```
load("Nimrod.rda")
x = table(Nimrod$medium)
x
pie(x, labels = c("Brass band","Concert band","Organ",
                  "Orchestra"),
  col = c("gold3","deepskyblue","peachpuff3","magenta"))
```

F.18　练习11-1

```
load("Nimrod.rda")
den = density(Nimrod$time)
plot(den)
rug(Nimrod$time)

library(nlme)
boxplot(MathAchieve$SES,main="Socioeconomic Status", ylab="SES
  score")
rug(MathAchieve$SES)
```

第一张图可能有些用处。第二张图太密集了，我们在大部分的数据范围内都无法看到点的间隔。

F.19　练习12-1

```
Year = c(2004:2010)
Europe = c(7.9, 7.9, 7.9, 7.8, 7.7, 7.1, 7.2)
Eurasia = c(8.5, 8.5, 8.7, 8.6, 8.9, 8, 8.4)

plot(Year, Europe, type = "l",
  col = "maroon", ylim = c(7,9),
  ylab = "Emissions", lwd = 2)
# ylim使图片足够大,因为Eurasia的值>7.9

lines(Year, Eurasia,
  lty = "dashed",
  col = "steelblue", lwd = 2)
legend("topleft", c("Eurasia", "Europe"),
  text.col = c("steelblue","maroon"),
  lty = c("dashed","solid"))
```

F.20 练习12-2

```
library(Sleuth2)
library(epade)
attach(case0701)
plot(Velocity, Distance)
```

正如所料，距离越远，速度越快。

```
scatter.ade(Velocity, Distance, wall = 3,
  main = "The Big Bang", col="red")
```

好极了！

F.21 练习13-1

```
library(nlme)
attach(MathAchieve)
plot(SES, MathAch)
```

这里可能有近似线性的关系，但我们很难确定。

```
library(nlme)
attach(MathAchieve)
sunflowerplot(SES, MathAch)
```

这张向日葵图并没有太大用处。

```
library(nlme)
attach(MathAchieve)
library(hexbin)
plot(hexbin(SES, MathAch), colramp = heat.ob)
```

我们所关注的关系更清晰了，图形几乎是垂直的；也就是说，对于所有 −1~1.5 的 SES 得分，有一个相同宽度的数学成绩范围。

```
detach(MathAchieve)
```

F.22 练习14-1

```
library(ResearchMethods)
load("baplot") # 如果保存了baplot
data(MFSV)
attach(MFSV)
baplot(MF,SV)
```

这里没有明显的模式，仅根据帮助文件中给出的信息，我们无法界定在临床上可以接受的差异。

F.23　练习15-1

```
library(reshape2)
library(car)
attach(tips)
tips$ratio = 100*(tip/total_bill)
attach(tips)
qqnorm(ratio)
qqline(ratio)
```

ratio 不是正态分布，它上端是偏斜的。

```
qq(time ~ ratio)
```

lunch 和 dinner 的分布截然不同。

```
qq(smoker ~ ratio)
```

smoker 和 nonsmoker 的分布也截然不同。

F.24　练习15-2

```
library(reshape2)
library(car)
attach(Vocab)
qqnorm(vocabulary)
qqline(vocabulary)
```

vocabulary 似乎并不是正态分布。

```
qqnorm(education)
qqline(education)
```

education 也不是。

```
qq(sex ~ vocabulary)
```

female 占据了大部分范围，male 在上限位。

F.25　练习16-1

```
library(car)
attach(Ginzberg)
head(Ginzberg)
pairs(~ simplicity + fatalism + depression)

scatterplotMatrix(~ simplicity + fatalism + depression)
```

```
library(corrplot)
corrplot(y)

library(GGally)
ggscatmat(Ginzberg, columns = 1:3)
```

F.26　练习17-1

```
library(scatterplot3d)
library(epicalc)
data(SO2)
scatterplot3d(SO2$smoke, SO2$SO2, SO2$deaths,
  highlight.3d = T,
  type = "h")
```

这是一个棘手的数据集，因为数据集和变量名称相同。尝试一下，你会发现，`attach(SO2)` 和 `scatterplot3d(smoke, SO2, deaths)` 无法工作。因此，这里要用 `SO2$smoke` 作为替代名称。死亡率似乎和其他变量正相关。

F.27　练习17-2

```
library(lattice)
library(epicalc)
data(SO2)
levelplot(SO2$deaths ~ SO2$smoke + SO2$SO2)
```

F.28　练习18-1

```
library(Sleuth2)
attach(ex1123)
plot(SO2, Mort)
```

除了一个高死亡率的点之外，二氧化硫（SO2）和死亡率（mortality）之间有很强的关系。

```
coplot(Mort ~ SO2 | Educ)
```

这个条件图表明，在人口教育水平最高的城市里，二氧化硫很少，死亡率较低。也许在拥有主要的金融企业或研究中心的城市中，烟囱较少。

F.29　练习19-1

```
attach(mtcars)
cars = as.matrix(mtcars)
h = dist(scale(cars))
h2 = hclust(h, method = "single")
plot(h2)
```

```
h = dist(scale(cars), method = "manhattan")
h2 = hclust(h, method = "single")
plot(h2)
```

F.30　练习19-2

```
cars = as.matrix(mtcars)
image(scale(cars), col = cm.colors(256))
image(scale(cars), col = rainbow(100))
image(scale(cars), col = terrain.colors(16))
```

诸如此类，colors 后边的数字表明范围内有多少种颜色。

F.31　练习19-3

```
install.packages("Amelia", dependencies = T)
library(Amelia)
missmap(airquality)

library(epade)
missiogram.ade(airquality)
```

F.32　练习20-1

维基百科中有介绍马赛克图的文章（https://en.wikipedia.org/wiki/Mosaic_plot），其中包括了 Titanic 数据集的示例。也可以访问 http://stackoverflow.com/questions/20228326/mosaic-plot-with-labels-in-each-box-showing-a-name-and-percentage-of-all-observa，找到该数据集不同方向的马赛克图。

F.33　练习21-1

```
attach(Nimrod)
par(bg = "white", mfrow = c(2,2))

# 图1
x = table(medium)
barplot(x, horiz = T,
  main = "Number of ensembles of each medium",
  names = c("Brass band", "Concert band", "Organ","Orchestra"),
  las = 1, cex.names = .8, col = "turquoise", space = 1.5)

# 图2
cols = "cadetblue4"
boxplot(time ~ medium,
  col = c("goldenrod","firebrick"),
```

```
  ylab = "Time", xlab = "Medium",
  varwidth = T,
  main = "Performance times by medium",
  col.main = cols,
  col.axis = cols,
  las = 1, col.lab = cols,
  names = c("","","",""))
mtext(text = c("Brass band", Concert band",
  "Organ", "Orchestra"),
  side = 1, cex = .6,
  at = c(1,2,3,4), line = 1)

# 图3
pro = subset(Nimrod, level == "p")
am = subset(Nimrod, level == "a")
plot(density(pro$time), ylim = c(0,.028),
  main = "Professional vs. amateur groups",
  xlab = "Time in seconds",
  col = "navy", lwd = 2,
  bty = "n", xlim = c(100,350),
  family = "HersheyScript",
  cex.main = 1.4, cex.lab = 1.3)
lines(density(am$time), lty = "dotted",
  col = "lightblue4", lwd = 2)
legend("topright", c("Amateur","Professional"),
  cex = .8, text.col = c("lightblue4","navy"),
  bty = "n")

# 图4
data2 = Nimrod[order(Nimrod$time),]
dotchart(data2$time,
  labels = data2$performer, cex = .34,
  main = "Performance Time by Performer",
  xlab = "Performance time", pch = 19,
  col = c("violetred1", "violetred4"),
  lcolor = "gray90",
  cex.main = 1.9,
  col.main = "violetred4",
  cex.lab = 1.4,
  family = "HersheySerif")
detach(Nimrod)
```

附录G

故障排查：为什么我的代码不工作

初学者几乎做什么都会犯错误。幸运的是，R 初学者犯错误时会收到报错信息，得到修改提示。遗憾的是，这些消息往往含糊不清，语焉不详。只要在 R 方面有一定的经验，你就可以很快解决这个问题。本附录列出了一些容易出现的错误以及相关的报错信息。这些都是我曾经犯过的错误，有些错误我还犯了不止一次。

G.1 拼写错误

拼写错误（misspelling）是最常见的错误之一。然而，R 的词库中显然没有 misspelling 这个词，因此这个问题会以其他多种方式标注，如下所示：

```
> attach(trees)
> plt(Girth, Height)
Error: could not find function "plt"
```

在这个例子中，R 提示称我们想使用的函数并不存在，或者处于隐藏状态，其实这里只是拼写出现了错误！重新输入拼写正确的命令，如下所示：

```
> plot(Girth, Height)
```

实际上，你不需要重新输入整个命令。上面这个例子的问题并不严重。不过，你有时会用到很长的命令，重复输入是相当麻烦的。在这种情况下，你有两种快捷方式可以选择。

- 复制并粘贴错误行到最新的提示符（>），修正错误，然后按下回车（Return 或 Enter）。
- 按向上键，复制前面一行命令。在这里，你还可以进行编辑。按向上键两次，你将复制上方两行命令。

另外一个例子如下所示：

```
> plot(Girth, Height, color = "red")
Warning messages:
1: In plot.window(...) : "color" is not a graphical parameter
2: In plot.xy(xy, type, ...) : "color" is not a graphical
   parameter
3: In axis(side = side, at = at, labels = labels, ...) :
  "color" is not a graphical parameter
4: In axis(side = side, at = at, labels = labels, ...) :
  "color" is not a graphical parameter
5: In box(...) : "color" is not a graphical parameter
6: In title(...) : "color" is not a graphical parameter
```

这组消息看起来很可怕，但仔细看每一行，你会发现 R 只是不能执行通过我们使用的单词 `color` 罢了。要找到正确的缩写，输入 **`?plot`**。如下所示，只需把 `color` 改为 `col`，程序就能很好地运行了：

```
> plot(Girth, Height, col = "red")
```

再看下一个例子：

```
> library(poltrix)
Error in library(poltrix) : there is no package called 'poltrix'
```

确实，这里没有名为 `poltrix` 的包。检查拼写！

```
> library(plotrix)
```

现在没有问题了！注意，本节中三个例子的问题都是拼写有误。R 却给出了三种完全不同的响应。这说明错误信息不一定要从字面上理解。要了解这些消息的真正含义，你需要一点经验。

G.2　令人困惑的大写/小写

记住，R 认为相同字母的大写和小写完全不同。在下面这个例子中，R 找不到 `height` 对象，因为没有这样的对象：

```
> attach(trees)
> plot(Girth, height)
Error in xy.coords(x, y, xlabel, ylabel, log) : object 'height'
not found
```

把 `height` 改为 `Height`，命令就可以运行了。如果你记不住准确的变量名，可以使用 `str(trees)` 命令或者 `head(trees)` 进行查找。

G.3 太少（或太多）的圆括号

在下面的例子中，R 不执行命令，并且打印出“+”而不是“>”：

```
> plot(density(Girth)
+
```

这意味着 R 在等你输入更多信息。在这种特殊情况下，在“+”之后再输入一个右括号即可让程序运行。

下面这个例子中圆括号太多。错误信息指向罪魁祸首：

```
> plot(density(Girth)))
Error: unexpected ')' in "plot(density(Girth)))"
>
```

使用带括号的表达式时，左括号的数量必须等于右括号的数量。如果输入的命令很长，在你按下回车键之前，最好数数左括号和右括号的数量。

G.4 忘记加载包

记住，如果不在基础 R 中却打算在会话中使用你安装的附加包，那么需要事先加载。假设你想对 `HistData` 包中的 `Nightingale` 数据集执行数据分析，却收到了下面这条信息：

```
> attach(Nightingale)
Error in attach(Nightingale) : object 'Nightingale' not found
>
```

这与拼写错误产生的错误信息相同，但二者实际上完全不同。此处的解决方案是首先加载 `HistData`，代码如下：

```
> library(HistData)
> attach(Nightingale)
```

这样一来，在退出 R 之前，你就可以使用 `HistData` 中的任意命令或数据集，而不用再次加载了。下次开启 R 时，如果想使用 `HistData` 中的内容，你还需要再次加载它。

G.5 忘记安装包

本书中很多例子使用 `library()` 命令加载包。记住，在加载包之前，必须使用 `install.packages()` 函数安装它，否则将会看到如下信息。

```
> library(abctools)
Error in library(abctools) :
  there is no package called 'abctools'
```

包只需要安装一次，安装之后会永远驻留在你的计算机上。然而，你必须在需要它的会话中加载它。

输入前面示例中的命令时，我还没有在计算机上安装 `abctools` 包。我可以使用如下命令安装它：

```
> install.packages("abctools")
```

使用下面这个命令会更好：

```
> install.packages("abctools", dependencies = TRUE)
```

这个命令不仅安装了 `abctools`，也安装了 `abctools` 可能依赖的任意其他包，即 `abctools` 本身使用的包。在基于 Windows 的计算机上，你还可以在“包”（Packages）菜单中选择“安装包”（Install Package(s)），以此完成安装。打开 CRAN mirror 窗口，选择一个镜像网站或者使用一个已加高亮的网站并点击 OK，然后选择你想要的包即可。Mac 用户可以进入“包和数据”（Packages & Data）菜单，选择“包安装程序”（Package Installer），单击“获取列表”（Get List）按钮，然后选择一个包。

除了最初的几章，本书在 `library()` 出现时都不会提醒你安装包。因为 `library()` 出现了很多次，所以我假设你知道需要做哪些准备。如果你不确定以前是否已经安装了包，可以使用如下命令查看你安装了哪些包：

```
> installed.packages()  # installed,不是install
```

这个命令提供的信息可能超出你的需要，它可以在所有 R 系统中运行。在基于 Windows 的机器上，还有一种便捷方式可以查看现有包，就是进入“包”（Packages）菜单并选择“加载包”（Load Packages）。Mac 用户可以进入“包和数据”（Packages & Data）菜单并选择“包管理器”（Package Manager）。

G.6　在加载的包中未找到数据集

加载完毕之后，包中的数据集应该是可用的，如下所示：

```
> library(reshape2)  # 加载包
> head(tips)         # 使用包中数据集

  total_bill  tip    sex smoker day   time size
1      16.99 1.01 Female     No Sun Dinner    2
2      10.34 1.66   Male     No Sun Dinner    3
3      21.01 3.50   Male     No Sun Dinner    3
4      23.68 3.31   Male     No Sun Dinner    2
5      24.59 3.61 Female     No Sun Dinner    4
6      25.29 4.71   Male     No Sun Dinner    4
```

上面的程序运行正常。但有时这里会出现问题，使得 R 找不到数据，如下所示：

```
> library(epicalc)
Loading required package: foreign
Loading required package: survival
Loading required package: splines
Loading required package: MASS
Loading required package: nnet
Warning message:
'.find.package' is deprecated.
Use 'find.package' instead.
See help("Deprecated")

> head(Ectopic)
Error in head(Ectopic) : object 'Ectopic' not found
```

`Ectopic` 是 `epicalc` 包中的数据集。包刚被加载，R 却提示它找不到该数据集。发生这种情况时，你可以再加一步，调用 `data()` 功能，加载你想要的特定数据集，代码如下：

```
> data(Ectopic)
> head(Ectopic)
  id outc      hia gravi
1  1 Deli  ever IA   1-2
2  2 Deli  ever IA   3-4
3  3 Deli never IA   1-2
4  4 Deli never IA   1-2
5  5 Deli never IA   1-2
6  6   IA  ever IA   1-2
```

这样就没问题了。

G.7　遗漏一个逗号

逗号会告知 R 一个参数在哪里结束，下一个参数从哪里开始。遗漏一个逗号会发生什么呢？来看看下面这段代码：

```
> plot(Girth, Height col = "red")
Error: unexpected symbol in "plot(Girth,Height col"
```

应当出现逗号的地方不能出现任何其他符号。参数必须由逗号分隔，如下所示：

```
> plot(Girth, Height, col = "red")
```

G.8　复制粘贴错误

下面的命令从会话中早些时候执行成功的命令复制而来。错误信息可能会使你摸不着头脑：

```
> > head(Nightingale)
Error: unexpected '>' in ">"
```

此处的问题是，该副本包括提示符号“>”，于是在粘贴之后，该命令行有两个提示符。这个问题有两种解决方法。

- 只复制“>”之后的命令行部分。
- 在粘贴之后，按下回车键之前，删除“>”。

G.9 目录问题——不能加载保存的文件

你可能发现自己先前保存的文件无法检索出来。即使拼写正确，你也可能找不到文件。此处可能出现这样的错误信息：

```
cannot open the connection
```

或者这样的错误信息：

```
no such file or directory
```

这很可能是因为你搜索的目录不对。换句话说，该文件不在你的工作目录中。

首先，请阅读 1.7 节。如果这部分没能解决你的问题，那就打开工作目录，看看你要找的文件是否真的存在。在基于 Windows 的计算机上，你可以进入“文件”（File）菜单，选择“显示文件”（Display file(s)）。在 Mac 上，你可以进入“文件”（File）菜单，选择“打开文件”（Open Document）。

不管是在那个平台上，只要文件不在工作目录中，你就需要搜索它，然后执行以下操作之一。

- 把它移动到工作目录中。
- 使用 `setwd()` 命令把工作目录改为放置文件的目录。
- 在 `load()` 命令中给出文件的完整路径。例如，如果需要的 cands.rda 文件在目录 /Users/yourname/Desktop/R Things/ 中，你可以使用如下命令。

```
load("/Users/yourname/Desktop/R Things/cands.rda")
# 注意引号
```

这里一定要确保拼写正确，斜杠、引号等符号也要输入正确。

G.10 丢失文件扩展名

这其实不是 R 的问题，但很容易给 R 用户带来问题。细想在 1.8.3 节创建的数据集，那是以电子表格输入的，以 CSV 文件导出。根据你的选择，在基于 Windows 的计算机或者 Mac 上，文件名可能显示为 Nimrod.Tempo。

这是因为你的计算机设置为不显示文件扩展名。如果我们记住的是这个文件名，发出的是带有此文件名的命令，那么 R 会表示这样的文件不存在，如下所示：

```
> Nimrod <- read.csv("Nimrod.Tempo", header = TRUE)
Error in file(file, "rt") : cannot open the connection
In addition: Warning message:
In file(file, "rt") :
  cannot open file 'Nimrod.Tempo': No such file or directory
```

简单地给文件名添加扩展名也没问题，如下所示：

```
> Nimrod <- read.csv("Nimrod.Tempo.csv", header = TRUE)
```

如果丢失文件扩展名 .R，尝试获取脚本时，你可能会遇到一个类似的问题，如下所示：

```
> source("NimTotals")
Error in file(filename, "r", encoding = encoding) :
  cannot open the connection
In addition: Warning message:
In file(filename, "r", encoding = encoding) :
  cannot open file 'NimTotals': No such file or directory
```

添加扩展名后，如下代码可以成功运行：

```
> source("NimTotals.R")      # 运行良好!
```

G.11　不要假定所有的包都使用相同的参数缩写

尽管很多包和基础 R 包在参数和缩写使用方面是一致的，但也有不一致的情况，比如下面这个例子：

```
> scatterplot3d(Solar.R, Ozone, Wind, col = "blue")
Error in scatterplot3d(Solar.R, Ozone, Wind, col = "blue") :
  argument 4 matches multiple formal arguments
```

参数 `col` 几乎是 R 包中的通用标准，但查看 `scatterplot3d` 的帮助文件，你会发现该包不能选择这个参数，而是需要使用以下参数：

```
> scatterplot3d(Solar.R, Ozone, Wind, color = "blue")
```

G.12　过时的包/包不兼容

许多包需要其他包处于可用状态。如果一个包无法加载，那可能是因为它所依赖的包不存在。如果有一个错误消息显示这类问题，例如点名指出了一个无法找到的包或命令，那么你也许只需要安装所需的包。有时，问题可能在于安装包的版本。以下命令可以确认这一点：

```
> library(help = packagename)
```

例如，如果加载 Rcmdr 时遇到困难，可以输入如下命令：

```
> library(help = Rcmdr)
```

你会看到关于该包的基本信息文件，其中包括需要的 / 建议的包和版本。下面是该文件作者命名后边的一小部分：

```
Depends:          R (>= 3.0.0), grDevices, utils, splines,
                      RcmdrMisc (>= 1.0-2), car (>= 2.0-21)
Imports:          tcltk, tcltk2 (>= 1.2-6), markdown, knitr,
                      abind
Suggests:         aplpack, colorspace, effects (>= 3.0-1),
                     e1071, foreign, grid, Hmisc, lattice,
                     leaps,lmtest, MASS, mgcv,
                     multcomp (>= 0.991-2), nlme, nnet,
                     relimp, rgl, rJava, RODBC, sem
                     (>= 2.1-1)
```

这段摘录表明，安装在这台计算机上的 Rcmdr 的版本要求 R 的版本为 3.0 或更高。这和已经安装其他几个包也有关系，比如 grDevices、utils，等等。此外，某些软件包必须是特定的版本，例如 car 必须是 2.0-21 或更高版本。如果没有匹配正确，那么 Rcmdr 就无法加载或无法正常工作。解决此问题最简单的方法就是重装 Rcmdr 及其依赖包，如下所示。

```
> install.packages("Rcmdr", dependencies = TRUE)
```

这样一来，Rcmdr 及其依赖包的最新版本就安装好了。如果仍然不能正常运行，你可能需要重新安装 R。

包还会引入帮助文件，比如对 Rcmdr 的 GUI 来说十分关键的 tcltk。如果没有这个包，那么你应该重新安装 R。此包通常是 R 的一部分，你需要确保安装的是二进制版本。如果你的计算机采用的是 Mac OS X 10.9 或更高版本，那么还需要安装 XQuartz。

最后，有一些包是建议安装的。没有这些包，Rcmdr 只有一部分可以正常工作。

附录H

本书介绍的R函数

本附录列出本书所用到的函数，以便于快速查询。这并不是完整的 R 函数列表。要获取某函数 *x* 的更多信息，请查看索引，或者输入 **`help(x)`** 或 **`?x`** 进行查询。

H.1　数据输入/输出

`load()`

　重新加载之前用 `save()` 创建的 R 数据集。

`open.ncdf()`

　打开 .ncdf 文件（`ncdf` 包）。

`Quandl()`

　读取 Quandl 数据集（`Quandl` 包）。

`read.csv()`

　读取 .csv 格式的文件并创建数据框。

`read.fwf()`

　读取固定宽度的文件。

`read.spss()`

　读取 SPSS 数据集（`foreign` 包）。

`read.table()`

　读取表格格式的文件并创建数据框。

`read.txt()`

读取 .txt 格式的文件并创建数据框。

`read.xport()`

读取 SAS XPORT 文件（`foreign` 包）。

`readWorksheetFromFile()`

读取 Excel 电子表格（`XLConnect` 包）。

`save()`

把 R 对象写入工作目录或者指定文件。

`xmlToDataFrame()`

读取 XML 文件（`XML` 包）。

H.2　数据集

`attach()`

为接下来的分析选择特定的数据集。

`data()`

确定有哪些可用的（无参数）数据集或者加载数据集。

`data.frame()`

合并两个或多个向量，生成一个新的数据框。

`detach()`

取消选定的数据集，也就是说，接下来的命令不再分析这个数据集。

`edit()`

编辑数据集。

`fix()`

编辑数据集。

`head()`

查看从数据集顶部选择的行。

`str()`

确定对象的结构。

`subset()`

创建一个指定数据框的子集。

`tail()`

查看从数据集底部选择的行。

H.3 绘图函数 1——创建图表

`barplot()`

生成条形图。

`bland.altman.ade()`

生成 Bland-Altman 图（`epade` 包）。

`boxplot()`

生成箱线图。

`coplot()`

生成协同图（条件图）。

`cor.plot()`

生成相关性分析图（`psych` 包）。

`corrplot()`

生成相关性分析图（`corrplot` 包）。

`dotchart()`

生成点图。

`dotplot.mtb()`

生成如统计软件 Minitab 中的点图（`plotrix` 包）。

`ehplot()`

生成 EH 图（`plotrix` 包）。

`fan.plot()`

生成扇形图（`plotrix` 包）。

`ggpairs()`

生成广义的对图（`GGally` 包）。

`ggplot()`

生成多种类型的图（`ggplot2` 包）。

`ggscatmat()`

生成顶部带有相关系数的散点图矩阵（`GGally` 包）。

`gpairs()`

生成广义的对图（`gpairs` 包）。

`grid()`

在现有图上绘制网格。

`heatmap()`

生成热图。

`heatmap.2()`

生成增强的热图（`gplots` 包）。

`hexbin()`

生成 hexbin 图（`hexbin` 包）。

`hist()`

生成直方图。

`Hist()`

为多个组生成一张直方图（`RcmdrMisc` 包）。

`histbackback()`

生成背对背直方图（`Hmisc` 包）。

`histogram()`

为多组生成直方图（`lattice` 包）。

`histStack()`

生成堆叠直方图（`plotrix` 包）。

`image()`

创建热图。

`levelplot()`

生成伪色图（`lattice` 包）。

`missiogram()`

生成缺失值的图（`epade` 包）。

`missmap()`

生成缺失值的图（`Amelia` 包）。

`mosaic()`

生成马赛克图（`vcd` 包）。

`mosaicplot()`

生成马赛克图。

`pairs()`

创建散点图矩阵。

`pie()`

生成饼图。

`pie3D()`

生成三维饼图（`plotrix` 包）。

`plot()`

生成散点图或其他图。

`PlotBubble()`

生成气泡图（`DescTools` 包）。

`pyramid()`

生成金字塔图（`epicalc` 包）。

`qq()`

生成分位数 – 分位数（QQ）图（`lattice` 包）。

`qqnorm()`

生成理论分位数的 QQ 图。

`qqplot()`

生成 QQ 图。

`scatter3d()`

生成带有回归面的三维散点图（`car` 包）。

`scatterplot()`

生成有高级特性的散点图（car 包）。

`scatterplot3d()`

生成三维散点图（scatterplot3d 包）。

`scatterplotMatrix()`

生成有高级特性的散点图矩阵（car 包），另一形式为 spm()。

`scatter.ade()`

生成有高级特性的散点图（epade 包）。

`smoothScatter()`

生成平滑散点图。

`spineplot()`

生成脊柱图（spinogram）。

`stem()`

生成茎叶图。

`stem.leaf()`

生成高级茎叶图（aplpack 包）。

`stripchart()`

生成带状图。

`sunflowerplot()`

生成向日葵图。

`xyplot()`

生成散点图（lattice 包）。

H.4　绘图函数 2——给现有图添加特征

`abline()`

在现有图上画一条直线。

`axis()`

为当前图添加坐标轴。

`legend()`

在当前图上添加图例。

`lines()`

在当前图上添加曲线。

`mtext()`

在当前图的边缘添加文本。

`par()`

设置或查询绘图参数。

`plotmath`

参见 `?plotmath`，在图上包含数学表达式。

`points()`

在当前图上画点。

`polygon()`

画 / 填充多边形。

`qqline()`

为 QQ 图添加一条线。

`rug()`

在当前图上画一张地毯图。

`text()`

在当前图的绘图区域输入文字。

H.5 其他

`asTheEconomist()`

模仿点阵图的样式（`latticeExtra` 包）。

`c()`

合并参数以生成向量。

`cat()`

打印函数的输出。

`colors()`

返回 R 的颜色名称。

`demo()`

演示选中的 R 的功能。

`dev.off()`

完成写入图形设备的操作，并保存文件。

`jpeg()`

打开以 .jpeg 格式保存的文件；必须以 `dev.off()` 结束。

`order()`

根据选定的变量的值，将数据框的行重新排序。

`par()`

设置或查询图表参数。

`png()`

打开以 .png 格式保存的文件；必须以 `dev.off()` 结束。

`print()`

打印输出。

`rgl.snapshot()`

将截图保存为 .png 文件（`rgl` 包）。

H.6 包

`available.packages()`

检查有哪些包可供下载。

`install.packages()`

下载并安装一个或多个 R 包。

`installed.packages()`

检查计算机上安装了哪些包。

`library()`

在当前会话中加载之前安装的包。

H.7 统计

R 有很多统计函数，本书没有一一介绍。我们提到过的函数包括下面这些。

`aggregate()`

把数据分成子集，计算每个子集的概括统计量。

`cor()`

计算皮尔森相关系数。

`CrossTable()`

以 SPSS 或 SAS 格式生成列联表（`gmodels` 包）。

`density()`

计算核密度估计。

`dist()`

计算矩阵行之间的距离。

`ecdf()`

计算经验积累分布函数。

`Ecdf()`

计算经验积累分布函数（`Hmisc` 包）。

`hclust()`

执行分层集群分析。

`lm()`

计算线性模型（比如回归模型）。

`max()`

计算向量的最大值。

`mean()`

计算向量的均值。

`median()`

计算向量的中位数。

`min()`

计算向量的最小值。

`quantile()`

寻找一个向量的分位数。

`scale()`

计算矩阵列的中心和 / 或规模。

`sd()`

计算向量的标准差。

`summary()`

计算向量的若干概括统计量。

`table()`

计算单向或双向频率。

`var()`

计算向量的方差。

H.8 用户自定义函数和脚本

`function() {}`

创建自定义函数。

`source()`

执行脚本。

H.9　工作空间和目录

`ls()`

确定当前目录中有哪些对象。

`getwd()`

查找当前工作目录。

`setwd()`

改变工作目录。

关于作者

John Jay Hilfiger，统计学家，数据分析专家，拥有生物统计学硕士学位，曾在三所重点高校（罗切斯特大学、爱荷华大学和康奈尔大学）担任统计 / 计算机分析员。他曾以副院长和主任的身份参与制度研究，充分利用了其数据方面的背景。此外，Hilfiger 还是一位作曲家和编曲家，发表过 100 多首作品，他曾是一名音乐教授，拥有音乐硕士和博士学位。他的大部分经历都涉及用数字、图形和洞见帮助他人分析数据，写作这本书也是这项事业的延伸。

关于封面

本书封面的动物是红黄拟啄木鸟（学名 Trachyphonus erythrocephalus）。它是 42 种䴕形目鸟类之一，分布于非洲东部国家（如肯尼亚和南苏丹），栖息地地形不平整，如河床、悬崖，以及白蚁巢穴或者土丘。

这种鸟外表独特、绚丽多彩。翅膀由黑色、红色和黄色组成，黑色部分散布着白色斑点（主要在翅膀和背部）。雄鸟和雌鸟头部大部分呈红色，雄性有黑色的额头和小羽冠。脖子和胸部为橙红色，下半身剩余部分为黄色。尾巴上有黑色和黄色的条纹。雌性和幼鸟比雄性颜色暗淡，并且橙色和红色部分比黄色和白色少。成年鸟身长可达 9 英寸，翼展 4 英寸，重约 72 克。

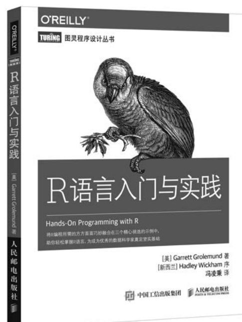
红黄拟啄木鸟是杂食动物，食物包括各种水果、种子和昆虫。它们也吃较小的鸟。由于所在的生存环境没有鲜明的季节变化，红黄拟啄木鸟只为获取食物而四处活动。

由于准备繁殖和养育幼雏过于辛苦，一对红黄拟啄木鸟往往需要一个或多个帮手。通常，这种鸟会在白蚁筑巢的地方挖一条隧道，巢室就在隧道尽头，由羽毛和草搭就。一巢一般有 2~6 个蛋。蛋孵化后，幼雏需要蛋白质，父母和帮手主要喂以昆虫。即便准备就绪，幼雏也不会离开鸟巢，而是会留下来帮助父母养育下一窝幼雏，直到该组建自己的家庭。

O'Reilly 封面上的许多动物都濒临灭绝，它们对于世界而言都弥足珍贵。想要了解如何提供更多帮助，可访问 animals.oreilly.com。

封面图片基于 *Cassell's Natural History* 的黑白版画，由 Karen Montgomery 着色而成。

延 展 阅 读



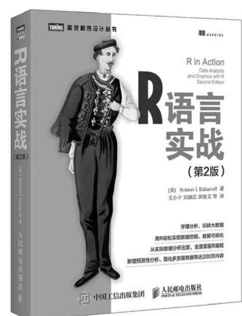
- 亲身实践三个数据分析项目
- 充分利用R的包系统和代码调试工具
- 在学习的过程中，实践和应用R的诸多编程概念

书号：978-7-115-42471-6
定价：59.00 元



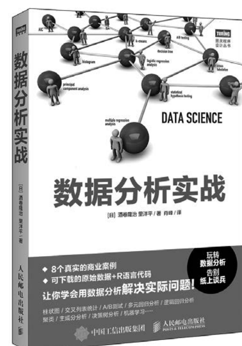
- RStudio首席科学家、R社区最有影响力的开发者Hadley Wickham详细展示如何开发R包、提高效率
- 统计之都创始人谢益辉、统计之都理事会主席冯凌秉作序推荐

书号：978-7-115-42788-5
定价：49.00 元



- 学懂分析，玩转大数据
- 用R轻松实现数据挖掘、数据可视化
- 从实际数据分析出发，全面掌握R编程
- 新增预测性分析、简化多变量数据等近200页内容

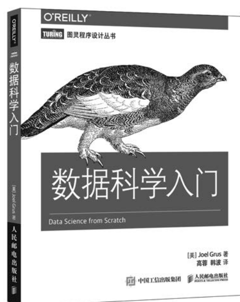
书号：978-7-115-42057-2
定价：99.00 元



- 通过8个真实的商业案例，让你学会用数据分析解决商业难题
- 使用未经清洗的原始数据，体验真实的数据分析流程
- 网罗柱状图、交叉列表统计、A/B测试、多元回归分析、逻辑回归分析、主成分分析、聚类、决策树分析、机器学习等数据分析方法
- 书中使用的数据和R脚本代码可下载

书号：978-7-115-45453-9
定价：45.00 元

延 展 阅 读



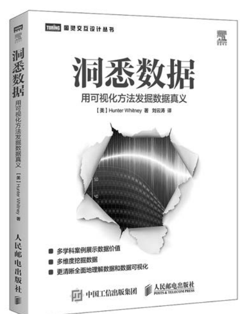
- 介绍数据科学基本知识的重量级读本，Google数据科学家作品
- 所有代码和数据均可在GitHub上下载

书号：978-7-115-41741-1
定价：69.00 元



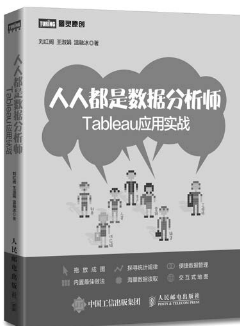
- 脱胎于哥伦比亚大学“数据科学导论”课程的教学讲义
- 大数据时代实战宝典，谷歌、微软、eBay等公司一线数据科学家真知灼见
- 揭秘数据科学相关的最新算法、方法与模型

书号：978-7-115-38349-5
定价：79.00 元



- 多学科案例展示数据价值
- 多维度挖掘数据
- 更清晰全面地理解数据和数据可视化

书号：978-7-115-41470-0
定价：69.00 元



- 简单易用，拖放成图，无需统计、计算机背景，即可进行可视化分析
- 分秒间读取，快速引擎处理，帮你看见并读懂大数据

书号：978-7-115-40686-6
定价：69.00 元

R图形化数据分析

与其在茫茫表格中搜索数字，不如将数据图形化，让复杂数据关系一目了然。本书是数据分析和可视化入门首选，以功能强大的R语言为工具，教你创建各种实用的数据图形，掌握高亮数据中的重要关系和趋势、简化数据形式、突出重点数字等技能。本书适合所有需要数据分析的读者，也可作为统计课程的补充教材，无需数学、统计学或计算机编程背景。

- **R语言基本知识**
- **创建单变量图，如饼图、箱线图、直方图等**
- **创建双变量图，如散点图、折线图、高密度图等**
- **创建多变量图，如散点图矩阵、三维图、树状图、热图等**

John Jay Hilfiger，统计学家，数据分析专家，拥有生物统计学硕士学位，曾在康奈尔大学、罗切斯特大学和爱荷华大学担任统计/计算机分析员。此外，Hilfiger还是一位作曲家和编曲家，拥有音乐硕士和博士学位，发表过100多首作品，曾任音乐教授。

DATA / DATA SCIENCE

封面设计：Karen Montgomery　张健

图灵社区：iTuring.cn
热线：(010)51095186转600

分类建议　计算机 / 数据库 / R

人民邮电出版社网址：www.ptpress.com.cn



O'Reilly Media, Inc.授权人民邮电出版社出版



ISBN 978-7-115-46441-5
定价：69.00元

看完了

如果您对本书内容有疑问，可发邮件至 contact@turingbook.com，会有编辑或作译者协助答疑。也可访问图灵社区，参与本书讨论。

如果是有关电子书的建议或问题，请联系专用客服邮箱:
ebook@turingbook.com。

在这可以找到我们:

微博 @图灵教育：好书、活动每日播报
微博 @图灵社区：电子书和好文章的消息
微博 @图灵新知：图灵教育的科普小组
微信 图灵访谈：ituring_interview，讲述码农精彩人生
微信 图灵教育：turingbooks